Evolving Optimal Feature Extraction using Multi-objective Genetic Programming: A Methodology and Preliminary Study on Edge Detection

Yang Zhang
Dept of Electronic & Electrical Engineering,
University of Sheffield
Mappin Building Mappin St. Sheffield S1 3JD, UK
+44(0)114-2225411
elp03yz@sheffield.ac.uk

Peter I Rockett
Dept of Electronic & Electrical Engineering,
University of Sheffield
Mappin Building Mappin St. Sheffield S1 3JD, UK
+44(0)114-2225589
p.rockett@sheffield.ac.uk

ABSTRACT

In this paper we describe a generic methodology to create an “optimal” feature extraction pre-processing stage for pattern classification. Our aim is to map the input data into a new, one-dimensional feature space in which separability is maximized under a simple thresholding classification. We have used multi-objective genetic programming with Pareto strength-based ranking to bias the selection procedure. The methodology is applied to the edge detection problem in image processing; we make quantitative comparison with the pre-processing stages of the well-known Canny edge detector using synthetic and real-world edge data and conclude that the performance of our evolutionary-based method is much superior to the Canny algorithm based on the criterion of minimum Bayes risk.

Categories and Subject Descriptors

Evolutionary Multiobjective Optimization

General Terms

Algorithms, Design, Theory.

Keywords

Edge Detector, Feature Extractor, Multi-objective Genetic Programming

1. INTRODUCTION

Feature extraction is a common and invaluable step in pattern classification. To utilize the potential information in a pattern set to its fullest extent is always a goal and to this end, subset selection, dimensionality reduction and transformation of features (feature extraction) have been applied to patterns before they are passed to a classifier. Whereas methods for dimensionality reduction and subset selection are well established [5], a generic and domain-independent methodology for feature extraction to minimize classification error has always been a highly desirable but so far, unattained goal in pattern recognition. Typically, feature extraction pre-processing approaches are hand-crafted based on domain-specific knowledge and optimality is hard to guarantee. Indeed, much of image processing research, for example, has been based on devising feature extraction algorithms.

Our objective in the present work has been to identify the optimal series of mathematical transformations to pattern data that produces the best class separation in the transformed feature space. Further, our aim has been to produce a generic, domain-independent method for generating optimal feature transformations such that the transformed patterns (or extracted features) can then be accurately classified with a simple and fast classifier.

Genetic programming (GP) is a problem-solving method [8] which can be viewed as evolving sequences of processing steps optimized with respect to some domain-specific fitness or objective. GP has already been used to develop automated feature detection/classification systems for pattern recognition tasks [18, 2, 3].

In [2], pipelined image processing operations were used to transform multi-spectral input data planes into a new set of image planes and a conventional supervised classifier used to classify the transformed features. The transformation stage used training data to derive a Fisher linear discriminant and then genetic programming was used to find a threshold to reduce the output from the discriminant-finding phase to a binary image. However, the discriminability is constrained in the discriminant-finding phase and the GP only used as a search tool in one-dimension to find a threshold.

In [4], the authors used GP to dynamically select feature subsets based on a few premises: Firstly, they focused the GP’s attention on the difficult cases (i.e. the ones which are frequently mis-classified, and cases which have not been looked at for several generations). They randomly selected a target number of cases from a training set every generation with a bias, hoping to attain one of the above two benefits. This approach, however, requires a very large training set which is often hard or expensive to obtain.

GECCO’05, June 25-29, 2005, Washington, DC, USA.

In the present work, we re-emphasize the importance to classification systems of the feature pre-processing stage, an area which has been overshadowed in recent years by advances in sophisticated classifiers (e.g. support vector machines). We propose evolving a transformation which projects input patterns into a one-dimensional feature space in such a way that class separability is "maximized. We have employed the multiple fitness objectives of: error rate (over a training set) together with a measure of tree complexity aimed at achieving the most compact mapping. Using this multi-objective optimization approach, we aim to produce a systematic and generic methodology for classification system design. In this paper, we describe the application of this methodology to the edge detection problem in image processing as a starting point since edge detection is a well-researched and understood field. Thus it is straightforward to gauge the usefulness of the evolved feature extraction method in comparison to conventional methods.

2. FEATURE EXTRACTION

Other researchers have used evolutionary methods to produce combined feature extraction *and* classification stages [7] but we argue that this approach makes the feature extraction procedure dependent on the co-evolved classifier in an completely opaque way. There is a potential risk with this combined approach that the evolved pre-processing can be excellent but the classifier can be poor giving a poor overall performance, or the pre-processing is limited to producing poor discriminability because the fitness is defined by a deficient classifier. In short, we argue that the search space in a combined feature extraction/classification method is too large and that evolving the feature selection method as a stand-alone processing stage is a more tractable undertaking. Classification has been very extensively studied and we see no merit in leaving the creation of an "optimal" classifier to evolutionary chance.

In the framework proposed here, feature extraction discovery is implemented as a supervised learning phase guided by the fundamental property of the misclassification error. We have applied this methodology to the edge detection problem in image processing to evolve an "optimal" mathematical transformation which maps the input pattern to a one-dimensional classification space in which we implement a decision making stage which is optimal with respect to the two class-conditioned PDFs in the decision variable. Given a set of feature transformations at any stage of the evolutionary process, we are always assured of the best pattern classification thus removing a sub-optimal decision making stage as a degrading factor in the overall system. This then allows us to concentrate the whole of our computational effort on evolving the best possible feature extraction stage.

Unfortunately, classification error alone is an insufficient fitness objective because the trees evolved by GP are well-known to 'bloat' – that is, grow excessively large with no corresponding improvement in performance. Consequently, in addition to misclassification error objective, we have also used a measure of tree complexity in a Pareto-based multi-objective optimization framework. In fact, our fitness vector comprises the *three* elements discussed in the following sub-sections:

2.1 Tree Complexity Measure

As pointed-out above, there is a danger that trees evolved by GP will become very large due to tree bloat. We have observed in early experiments that huge trees could produce a extremely small error over the training set but a very poor error estimated over an independent validation set. A similar phenomenon is seen in neural networks where an overly complex network over-fits the training set. The balance between training error and complexity is a recurring theme in classifier design (and other fields) and is often referred to as the bias-variance dilemma. Broadly, for a given training error, the simpler individual is preferred. Thus we have used node count as a straightforward measure of tree complexity to be one of our fitness vector elements driving the evolution to producing the smallest tree possible.

2.2 Bayes Error Estimate

The use appropriate fitness functions is critical to the search performance of genetic programming. An inappropriate fitness function can seriously mislead the evolution. Some previous work has been done aimed at searching for an optimal fitness evaluation method, for example [11], in which each individual in the population returns a real number and this is treated as if it was a prediction of the true percentage of the desired class. These authors used two components of fitness: squared error and (½ *TP* + ½(1-*FP*)). To calculate the true positive (TP) and the false positive (FP) rates, the value returned by a tree is truncated and compared with an experience threshold. The potential risk in this approach lies in the fact that they have used a crude classifier to estimate *TP* and *FP*, and the use of the equal priors (*Pr* = 0.5) as the second fitness component, which is not always be reasonable in real applications. Furthermore, they reuse the squared error as another component of fitness, but actually it is the average truncated error on the training set.

Directly and simply, we choose the most commonly used nonparametric estimator – a histogram – to make an empirical estimation of class-conditioned likelihood densities in the one-dimensional projected decision space. Considering these two class-conditioned densities, the Bayes error is estimated to evaluate the separability associated with a given feature extractor (population individual) by means of the class prior probabilities and calculating the overlap between the two densities. The Bayes error is a fundamental lower bound on classification performance, dependent solely on the class-conditioned densities and independent of the classifier. It yields a fast, simple and direct objective which seeks to minimize the overlap between the class-conditioned likelihoods.

2.3 Misclassification Error

The final element we use in the fitness vector is the conventional one of the fraction of misclassified patterns counted over the training set. Since we are projecting the input pattern to a one-dimensional decision space, here we use a simple threshold classifier. This objective means we are trying to evolve a feature extractor which maps the original pattern space into a new feature space where threshold classification would be able to yield the minimum possible misclassification.

The use of *two* measures of classification error – the Bayes error estimate (Section 2.2) and the fraction of misclassified training set patterns (Section 2.3) requires some explanation. In the early stages of this work it became clear that using only the fraction misclassified patterns resulted in slow and sometimes, non-convergent evolution. In the initial phases of the evolution where all the randomly generated members of the population were typically poor performers, the fraction of misclassified patterns objective lacked the sensitivity to identify individuals with slightly more promise than others, hence the slow convergence.

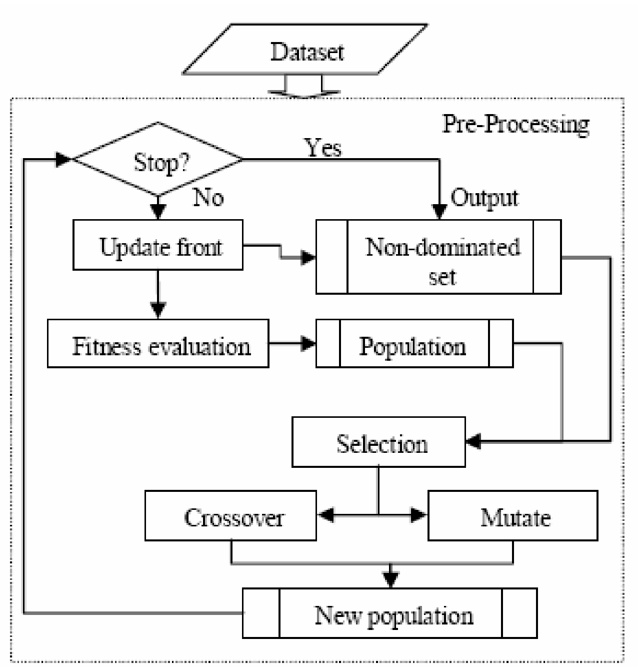


Figure 1 GP workflow

The use of the Bayes error alone allowed the evolutions to converge rapidly but the subsequently estimated *validation* error was very high. On closer inspection, it became clear that although the Bayes error objective was minimized over the training set, the GP was often opportunistically achieving this goal by producing two transformed class-conditioned densities with non-coincident 'comb'-like features rather than the desired end of two compact densities with widely separated means. Consequently, although the degree of likelihood overlap from the training set was small, the misclassification fraction calculated over the validation set was large. This led us to using two error measures: the Bayes error allows the evolutionary search to make rapid progress in the initial stages of the optimization while the fraction of misclassified patterns eventually comes to the fore when the evolution advances to a certain stage of maturity and leads to two well separated distributions.

2.4 GP Implementation

Our genetic programming implementation used a generational strategy in which two sets of individuals are maintained during evolution. One represents the current population and the second contains the current approximation to the Pareto front. The ranking is done by calculation of strength or fitness of each individual in both sets. When calculating the strength for individual we use the method designed in SPEA-2[10].

The workflow of our multi-objective genetic programming algorithm is illustrated at Figure 1.

Non-destructive, depth-dependent crossover [12] was used in our work in order to avoid the breaking of building blocks. After breeding, individual trees grow to be much bigger than the original ones and simple depth-dependent crossover did not work effectively. Retaining offspring if their raw fitness was better than their parents did not always work, especially as we are dealing with high-dimensional patterns in a large search space. It would be possible to work with big trees both in depth and in number of nodes. Furthermore, we found that simple non-destructive depth-dependent crossover does not work well when applied to multi-objective genetic programming.

Table 1 GP settings

Terminal	Pixels in 13 × 13 patch 10 Float number in {0.0,…,1.0}
Function	RADICAL, LOG, POWER, MINUS, SIN
	MINUS, PLUS, MULTIPLE, DIVIDE, MAX, MIN
	IFELSE
Raw fitness	Bayesian Error; Classification Error; Node Number
Standardized fitness	Strength-based Fitness
Population size	500
Original Tree Depth	5
Selection Method	Binary Tournament Selection
Max Generation	500
Stopping Criterion	best individual in generation does not evolve across 4% * Max of Generations or Max generations arrives or Error == 0
Operators	Crossover 70%, Mutation 30%
Nodes Type	UNARY, BINARY, TERNARY
Original population	Half Full tree, half random tree

Thus we modified the crossover operator to retain only the offspring which *dominates* either of its parents. In this way, we were able to maintain diversity in the population and avoid being trapped in local minima in the early stages. Most importantly, this crossover method works much more consistently in a multi-objective GP by avoiding comparisons of raw fitness.

We suggest and use a depth-dependent mutation operator in the current GP implementation. The contribution from mutation and crossover to effective search in a defined space has been discussed at length elsewhere. Here we chose a sub-tree of individuals to mutate based on uniform selection of depth and then reuse the strategy in the above crossover operator – we select a sub-tree based on its complexity.

The parameters used in the GP implementation are listed in Table 1.

3. DATA MODEL

In a study of training neural network edge detectors, Chen et al. [15] concluded that hand labeling real image data yielded a training set which did not adequately sample the pattern space leading to deficient learning. Consequently, we have used a similar approach of synthesizing both training dataset and validation dataset using a physically realistic model of the imaging process. Following [15], we have created three types of patterns: edges, non-obvious-non edges and uniform patches as illustrated in Figure 2a and 2b. Figure 2a depicts an edge configuration since the edge passes through the central pixel of the odd-sized patch. Figure 2b shows a non-obvious non-edge (NONE) since although the configuration represents an edge pattern, it does not pass through the central cell of the patch and thus we do not want such a pattern labeled as an edge.

We have modeled an abrupt step edge located on the optical axis, which is then projected onto the CCD image plane. A set of edge/NONE patterns were produced by applying randomized affine transformations of the optical field image projected onto the CCD. A uniformly-distributed random rotation angle in [0…2π] was applied followed by uniformly-distributed displacements:

$$\Delta x \in [-1.5 \ldots 1.5] \text{ pixels}$$

$$\Delta y \in [-1.5 \ldots 1.5] \text{ pixels}$$

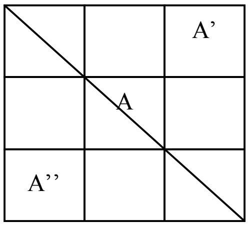


Figure 2a Edge pattern

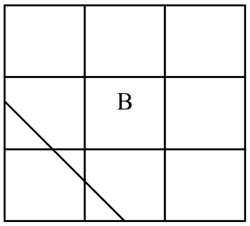


Figure 2b Non-obvious non-edge pattern

The intensities on either side of the step edge were randomly selected in [0…255]. Finally, a Gaussian blur (σ = 1) was applied to approximate the point spread function (PSF) of the imaging system. Further details can be found in [1].

Both the training and validation datasets used here were produced independently. We have employed an image patch size of 13 × 13 – probably larger than is needed to investigate whether GP can select the most useful features within the constraint of minimizing the tree size and by implication, the number of input features used. 10,000 training set patterns were generated, and 100,000 samples for the validation set maintaining a constant prior probability of the edge class of 0.05.

Every edge pixel is accompanied by two non-obvious-non edge pixels as relationship between A', A'' and A in figure 2a. So the prior probability for the non-obvious-non-edge is 0.1 in both training and validation sets. As a consequence of the foregoing, uniform patches comprise 85% in both datasets.

As a final stage in generating the *validation* set, zero-mean, independent, Gaussian-distributed noise (σ = 2) was added to every pixel. Note that the training set was *not* corrupted by noise since each training set pattern can be viewed as a 'mean' pattern of a large number of noise corrupted patterns; rather than presenting a large number of noise-corrupted patterns to the optimization, it is computationally faster to use noise-free 'mean' patterns.

4. RESULTS AND COMPARISIONS

Fair quantitative comparison of the results obtained here with other edge detection methods turns out to be a subtle and involved issue. The obvious conventional algorithm with which to make comparison is that due to Canny which is widely held to be the best edge detector currently available [9,13]. The Canny algorithm, however, includes a significant number of post-processing steps such as non-maximal suppression (NMS) which appear to be responsible, in large part, for the superiority of the Canny algorithm over other conventional edge detectors [16]. We contend that the most appropriate basis for comparison is the labeling point which minimizes the Bayes risk (for equal costs of error [5]) for both the Canny and GP detectors. However, arranging this for the GP detector developed here is not straightforward since this would require the post-processing steps to be included within the optimization loop to guarantee the minimum Bayes risk operating point with post-processing. Therefore, in order to compare like-with-like in the present work we have used the output of the optimal linear filtering stage from the Canny algorithm but *before* the post-processing. We argue that this is a valid basis for fair comparison because NMS and other post-processing steps can equally well be applied to the output of our GP detector to reduce the false positive rate. Comparison on an equal footing *before* post-processing exposes the fundamental quality of the edge detection method. Hereafter, we refer the "Canny" algorithm as that without the post-processing. Our interest here is a domain-independent methodology and steps like NMS are very much specific to the image processing domain.

We have carried out a number of runs to evolve individuals using a training set and compared the validation error using optimal "feature extractor" with the validation error from the Canny edge detector. The operating point which naturally emerges from our GP algorithm is that which minimizes the misclassification error. For the "Canny" algorithm we need to set the decision threshold which minimizes the Bayes risk and this can be obtained straightforwardly from the receiver operating characteristic (ROC) plot calculated over the validation set. The operating point which minimizes the Bayes risk is given by the tangent point of the line of slope, k [14, 17]:

$$k = \left(\frac{C_{TN} - C_{FP}}{C_{TP} - C_{FN}} \right) \times \left(\frac{1-P}{P} \right) \qquad \ldots(1)$$

where C_{TN}, C_{FP}, C_{TP}, C_{FN} are costs for true-negative, false-positive, true-positive and false-negative, respectively and P is the prior probability of edges (here, 0.05) [14]. We have used a cost ratio of one since in the edge detection problem we have no basis for regarding one sort of error as more or less important than another. The ROC plot for the "Canny" detector with the filter width set to the popular value of $\sigma = 1$ is shown in Figure 3 from which it is clear that the optimal operating point is very poor. The "Canny" error is 0.0496 which contrasts with the error of 0.05 which can be obtained by simply labeling every patch a non-edge without considering the input data at all. In fact, the main contribution to this error is the poor true positive (TP) figure.

The poor apparent performance of the "Canny" algorithm was something of a surprise here. Replotting the ROC curve using the ground truth labeled real image data from Heath et al. [9] produced almost identical results implying that the synthetically generated edge data is not responsible for the poor showing of the "Canny" algorithm. In fact, we have been unable to identify any previous work to determine the operating point for this detector which minimizes the Bayes risk, as opposed to finding a minimum misclassification rate [e.g. 9]. The fact that workers in the field routinely obtain 'sensible' edge maps from the Canny algorithm implies that users are sub-consciously imposing significant cost ratios (see eqn. 1) when they select an "optimal" threshold.

The corresponding misclassification error for the GP edge detector is shown in Table 2 along with the Bayes error estimate computed over the edge and non-edge likelihood distributions. We have calculated the confidence intervals at a 95% confidence level for both the Bayes error and the validation error and conclude that the differences between the GP and "Canny" detectors are statistically significant. Clearly the GP performance is superior. It is also clear that the "Canny" algorithm does not achieve particularly good class separation for case of a cost ratio of unity considered here.

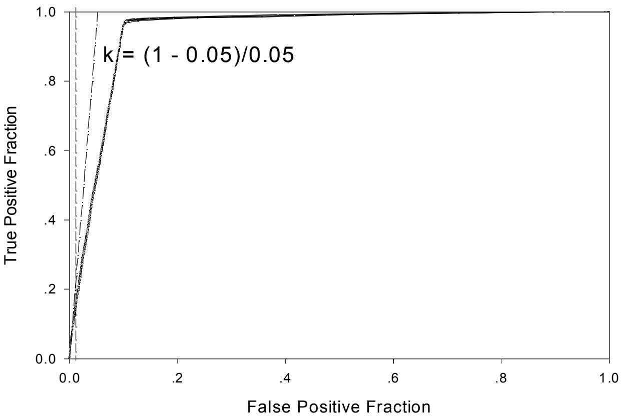


Figure 3. ROC plot for the "Canny" algorithm

Typical results from a range of optimization runs which display the smallest classification errors are shown in Table 3 from which it is clear that for many of the evolved trees, the misclassification error is quite close to the Bayes error estimates. This implies that the feature extraction sequences obtained are consistently close to being Bayes optimal.

By way of illustrating a typical GP tree obtained here, Figure 5 shows a tree obtained at generation 81 which contains 45 nodes. The Bayes error estimate for this tree is 0.028970 and the validation error is 0.029430.

Table 2 Comparison

Method	Bayes Error	Classification Error
GP	0.0248	0.0264
Canny	0.0489	0.0496

Table 3 Run list

Generation	Nodes	BE	Error
25	3	0.0203	0.03544
169	59	0.02472	0.02482
94	31	0.02899	0.02978
99	21	0.02242	0.02312
105	29	0.029	0.02928
73	38	0.02819	0.02876
39	99	0.02572	0.02606
62	93	0.02699	0.02805
90	60	0.01987	0.02189
110	30	0.02337	0.02398
52	28	0.0129	0.01291
109	34	0.03359	0.03552
88	110	0.02625	0.02664
85	120	0.02724	0.02752
136	52	0.02206	0.02226

In order to examine the labeling performance on real image data, we have applied the GP feature extractor shown in Figure 5 to images taken from the ground truth labeled USF dataset [9] and drawn comparison with the "Canny" edge detector. We have determined the optimal thresholds for the "Canny" detector for each image in turn from the ROC curves plotted for each image and using the appropriate edge prior for each test image. The labeling performance for minimum Bayes risk for the "Canny" detector is summarized in Table 4 from which it is clear that the poor performance of this algorithm on the validation dataset is repeated on real image data.

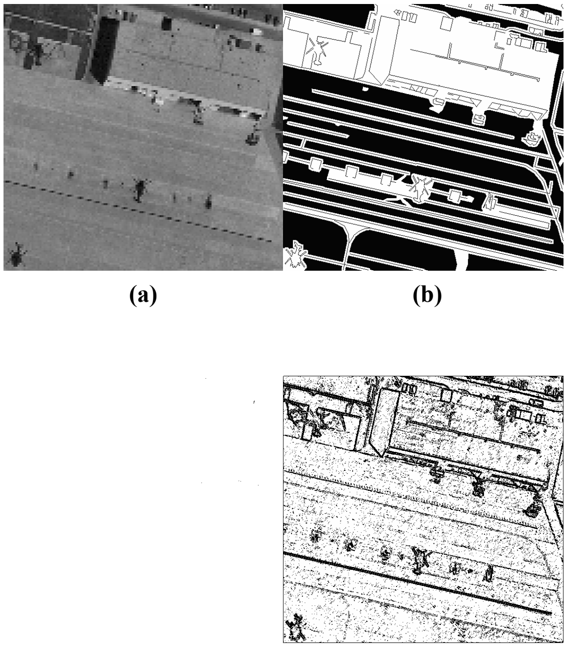
In Figure 4.1 to Figure 4.3, "a" denotes the original images from the USF dataset, "b" shows the ground truth data, "c" shows the labeling results from the "Canny" detector using image-specific "optimal" threshold and "d" shows images labeled with the GP feature extractor illustrated in Figure 5.

The comparisons in Figure 4.1 to 4.3 are made on the basis of adjusting the "Canny" threshold to give the minimum Bayes risk whereas the GP results were obtained for the feature extractor trained on a fixed prior of 0.05, i.e. not completely optimal with respect to the priors.

The most striking result is that the "Canny" algorithm fails to label almost all the edges; Figure 4.1(c) contains just six labeled pixels in the top right hand corner of the image.

The GP results on Figures 4.1 – 4.3 are summarized in Table 5 from which it can be seen that this detector attains a significantly higher true positive (TP) figure. The high false positive figure of the GP detector is consistent with the thick edges labeled in Figures 4.1(d), and particularly Figure 4.2(d) and Figure 4.3(d).

(Such this edges could be thinned by non-maximal suppression post-processing which has not been applied to these results.)


(a) (b)

(c) (d)

Figure 4.1 (a)-(d). See text for details

Table 4 Canny [TP, FP] operating points on test images

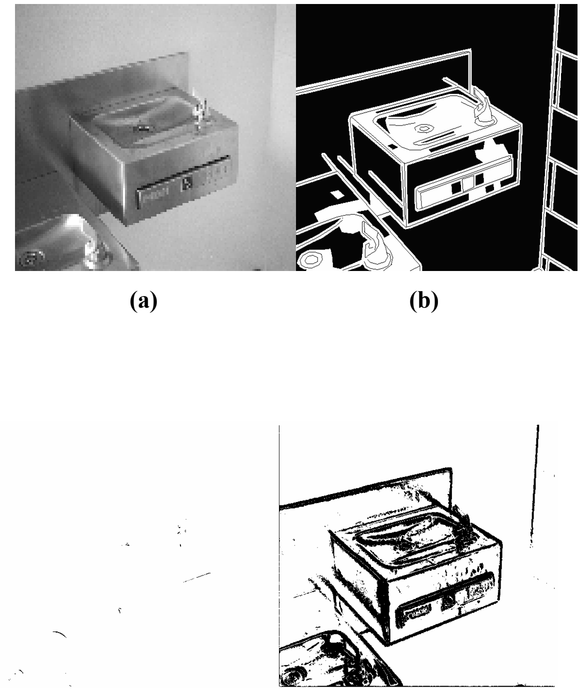
Figure	Edge Prior	TP	FP
4.1	0.043	0.00029	0.0001
4.2	0.025	0.0142	0.0063
4.3	0.034	0.0320	0.0045

Table 5 GP [TP, FP] operating points on test images

Figure	TP	FP
4.1	0.6295	0.1834
4.2	0.7600	0.1500
4.3	0.8807	0.3400

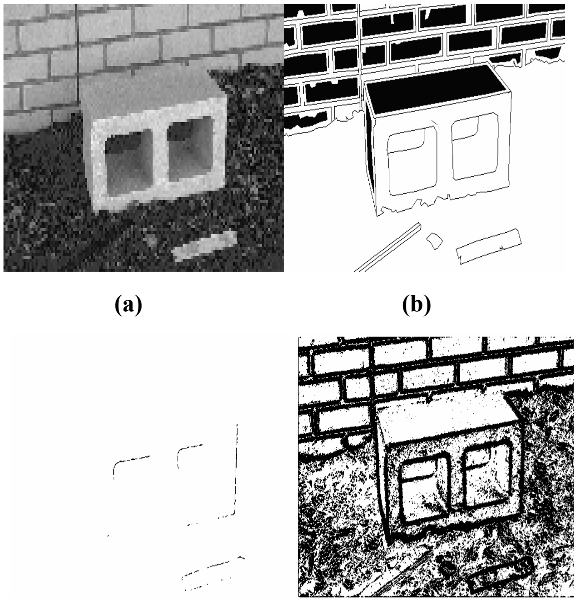
5. DISCUSSION AND FUTURE WORK

We have demonstrated that multi-objective genetic programming is able to evolve an “optimal” feature extractor to transform input patterns into a feature space in which we could obtain maximum separability. In the present work we have projected the input pattern to a one-dimensional decision space since this transformation naturally arises from a genetic programming tree. Potentially, superior classification performance can be obtained by projecting the input features to a multi-dimensional decision space and this is currently an area of active research.


(a) (b)

(c) (d)

Figure 4.2 (a)-(d).


(a) (b)

(c) (d)

Figure 4.3 (a)-(d).

The use of multiple objectives, particularly multiple classification error objectives has been shown to be effective in guiding the optimization. It is interesting that GP was able to meet its goals of minimizing the overlap of the two likelihoods in a way which was both unintended and unwanted. Clearly the straightforward concept of separabilty needs to be very carefully framed for GP to avoid evolving opportunistic and unhelpful solutions.

The node number objective employed penalizes an individual according to its complexity. This appears to be essential both in order to prevent tree bloat as well suppressing over-fitting of the training set leading to poor generalization.

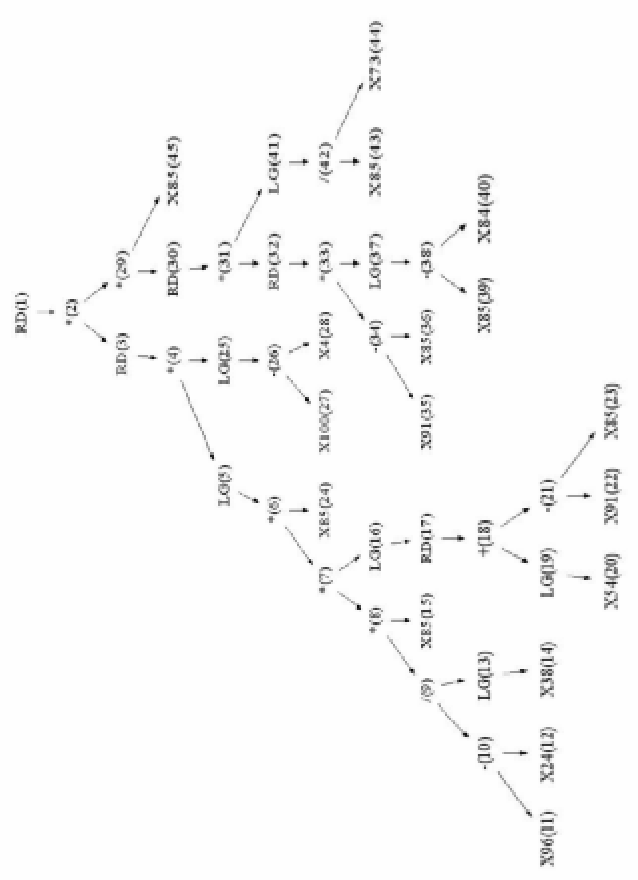


Figure 5 Typical GP individual

Although the present work was not intended as a feature selection method, we have presented the optimization with a 13 × 13 image patch. Since one of our simultaneous objectives was to minimize the node count – and by implication the number input features used – it is interesting to note that in all cases the GP selected the pixels from around the center of the image patch which is intuitively pleasing. One would expect most of the edge 'information' is located around the center of the patch.

Another area of ongoing work is the interpretation of evolved tree. Certainly the tree shown in Figure 5 is quite unlike any conventional edge detection algorithm although given that the Canny algorithm – widely held to be the best conventional edge detector performs so poorly in a like-for-like comparison - this is not surprising.

Finally, we plan to apply the present methodology to other classification problems such as corner detection in image processing as well as more general problems.

6. CONCLUSIONS

In this paper we have presented a domain-independent multi-objective genetic programming methodology to evolve near-optimal feature extraction algorithms. We have employed both an estimate of Bayes error and misclassification error to drive the optimization since the combination of these two results in faster convergence and better generalization performance.

As a demonstration vehicle, we have examined the problem of edge detection in image processing and made quantitative comparison with the pre-processing stages of the well-known Canny algorithm. One surprising (and incidental) outcome of this work has been that the operating point for the Canny algorithm which minimizes Bayes risk is very poor. In direct comparison with the Canny algorithm, the feature detector evolved using genetic programming performs significantly better.

7. REFERENCES

[1] P.I.Rockett. Performance Assessment of Feature Detection Algorithms: A Methodology and Case Study on Corner Detectors. *IEEE Transactions on Image Processing.* Vol.12, No.11. Nov 2003.

[2] N.R. Harvey, S.P. Brumby, S. Perkins, J.J. Szymanski. J. Theiler, J.J. Bloch, R.B. Porter, M. Galassi & A.C. Young. Image Feature Extraction: GENIE vs Conventional Supervised Classification Techniques. *IEEE Transactions on Geoscience and Remote Sensing,* Vol. 40, No. 2, pp 393-404, 2002.

[3] J.R.Sherrah, R.E.Bogner & A.Bouzerdoum. The Evolutionary Pre-Processor: Automatic Feature Extraction for Supervised Classification using Genetic Programming. *Genetic Programming 1997: Proceedings of the Second Annual Conference. Stanford University, CA, USA.* Pages 304-312 1997

[4] C.Gathercole & P.Ross, Dynamic training subset selection for supervised learning in genetic programming, *Parallel Problem Solving from Nature-PPSN III*, pp. 312-321, *Springer-Verlag*, 1994.

[5] K. Fukanaga. Introduction to Statistical Pattern Recognition. *Academic Press, Boston, second edition*, 1990.

[6] L.Devroye, L. Györfi & G. Lugosi. A Problabilistic Theory of Pattern Recognition. *Springer-Verlag*, 1996.

[7] J.R.Sherrah, Automatic Feature Extraction for Pattern Recognition. *PhD Thesis, Dept. of EEE, The University of Adelaide, South Australia,* July. 1998

[8] P.J.Angeline. Genetic programming and emergent intelligence. In *K.E.Kinear, Jr., editor, Advances in Genetic Programming,* chapter 4, pages 75-98. MIT Press, 1994.

[9] M.D.Heath, S.Sarkar, T.Sanocki & K.W.Bowyer. Comparison of Edge Detectors: A Methodology and Initial Study. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96,* Pages 143-148. 1996

[10] S. Bleuler, M. Brack, L. Thiele & E. Zitzler. Multiobjective Genetic Programming: Reducing Bloat Using SPEA2. Congress on Evolutionary Computation (CEC 2001). Pages 536-543. 5/2001

[11] W. B. Langdon & S. J. Barrett. Genetic Programming in Data Mining for Drug Discovery, Chapter 10 in Evolutionary Computing in Data Mining, Ashish Ghosh and Lakhmi C. Jain *editors*, Physica Verlag, pages 211-235, 2004.

[12] T. Ito, I. Iba & S. Sato. Non-destructive depth-dependent crossover for genetic programming. In W. Banzhaf, R. Poli, M. Schoenauer, and T. C. Fogarty, editors, *Proceedings of the First European Workshop on Genetic Programming*, LNCS, Paris, 14-15 April 1998.

[13] K.Bowyer, C.Kranenburg S.Dougherty. Edge Detector Evaluation Using Empirical ROC Curves. In *Computer Vision and Image Understanding* vol.84, NO.1, pages 77-103 Oct. 2001.

[14] M.H.Zweig & G.Campbell. Receiver Operating Characteristic (ROC) Plots: A Fundamental Evaluation Tool in Clinical Medicine. In *Clinical Chemistry*, vol. 39, iss. 4, pp561-577 1993

[15] W.C.Chen, N.A.Thacker & P.I.Rockett. An adaptive step edge model for self-consistent training of a neural network for probabilistic edge labeling. In *IEE Proceedings - Vision, Image & Signal Processing*, VISP 143 No.1, Pages 41-50 Feb. 1996.

[16] R.K.Cope & P.I.Rockett. The efficacy of Gaussian smoothing in the Canny edge detector. In *Electronics Letters*, Vol 36 (19), Pages 1615-1617 2000.

[17] T.Kanungo & R.M.Haralick. Receiver operating characteristic curves and optimal Bayesian operating points. In *International Conference on Image Processing - Proceedings,* Vol.3 Pages. 256-259, Washington, DC, Oct 23-26, 1995

[18] W.A.Tackett. Genetic Programming for feature discovery and image discrimination. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, Pages. 303-309. Morgan Kaufmann, 1999.