ADDING NEW FEATURES AND CLASSES TO CLASSIFIERS EVOLVED USING GENETIC PROGRAMMING

Panitnat Yimyam and Adrian F. Clark

School of Computer Science and Electronic Engineering, University of Essex, Colchester, UK

ABSTRACT

This paper considers the need to re-train a multiclass classifier that has initially been evolved using genetic programming to accommodate new features or new classes. For the former, the new feature is incorporated into a program by mutation; after that, a program that performs classification using all the features is obtained by evolution. For the latter, a binary classifier is evolved that is able to distinguish the new class from all existing classes is evolved, and it is executed before the existing classification programs. The two approaches are demonstrated on a range of classification problems drawn from the general area of produce grading and the results demonstrate the effectiveness of the proposed approach, in terms of both computational speed and classification performance.

Index Terms— Adding new features and classes, Genetic programming, Classification, Agricultural grading

1. INTRODUCTION

Genetic Programming (GP) is a machine learning technique used in many problem domains and has the attraction that it yields *programs* that can be executed to perform tasks. The research described in this paper is concerned with using GP to construct complete computer vision systems (i.e., feeding in images and obtaining classes of features) automatically from training data. Oechsle and Clark [1] demonstrated that this was possible, and more recently Yimyam and Clark [2] have shown that the performance achievable by a system trained in this way depends critically on the operators used to extract information from images. The underlying approach is to evolve a series of binary classifiers, each of which distinguishes one class of feature. In the early stages of the system, such a classifier operates on pixels to separate features from background, and subsequently classifiers are evolved to distinguish the various classes of feature. This approach has been found to produce systems automatically whose performance is commensurate with other machine learning approaches and with human-written solutions. The approach also scales, with [3] reporting the successful evolution of a system able to recognise over 30 classes.

Taking that work as a starting point, the research in this paper addresses two particular questions. Firstly, when a solution has already been evolved using features, is there some way in which a further feature can be accommodated? Secondly, if a system has been evolved to distinguish N classes, can an $(N+1)$th class be added? In both cases, extending the capability of the system needs to be done with less computation than evolving a new solution from scratch.

The remainder of this paper is structured as follows. Section 2 reviews the existing literature on extending GP-based classifiers. Section 3 details the approach followed in this work. Section 4 shows experimental results, and finally section 5 draws conclusions and identifies future directions for the research.

2. EXTENDING GP-BASED CLASSIFIERS

There are several reports of GP having been modified to produce better classifiers. [4] presented a new approach named *GP models for class discrimination* (GPMCD). They noticed that some characteristics of samples in one class were also present in others, so they evolved programs that combined strongly-related attributes into a unique pattern for each class and applied that to class discrimination. They compared the performance of their approach with linear discriminant analysis (LDA), k-nearest neighbour (kNN), classification and regression tree (CART), artificial neural networks (ANN), and support vector machines (SVM). The experimental results showed that GPMCD was able to achieve the most accurate results several times, especially in the discrimination of new samples.

[5] proposed a bundled-GP scheme. Binary classifiers were assembled into a bundle of classifiers for each class, and then they were combined to generate a multiclass classifier, simultaneously reducing the number of selected features. Both (tournament) selection and crossover had to be adapted to work with these classifier bundles. From the results, the bundle-GP method was a little more effective than conventional independent GPs and spent more time for selecting features; however, it was able to distinguish classes from selected features.

Some researchers have adapted the conventional GP fitness functions. For example, [6] performed experiments on object detection by using GP with a fitness function technique called *clustering based fitness* (CBF). It was found that

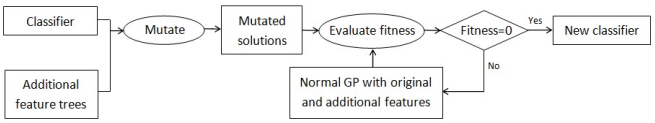

Fig. 1. Incorporating new features using mutation

their function yielded a large false alarm rate, so they refined it to be based on a relative localization weighted F-measure (RLWF) which was able to reduce false alarms, and required a substantially shorter training time than CBF. [7] proposed *constrained GP* with a 'constrained' fitness function calculated from the rate of TP and FP classifications.

Some methods have attempted to enhance the feature selection processes of GP [8, 9]. For instance, [10] proposed *modified Fisher linear discriminant analysis* (MFLDA) to help GP to evaluate solutions. Their results suggest that a GP classifier based on MFLDA was superior to other techniques.

This paper focuses on the binary decomposition approach addressed in [11] for multi-class problems. This decomposes the multi-class into a group of binary classifications, then generates a binary classifier for each class. From the experimental results, this approach was able to achieve higher accuracy on the muli-class data sets than several other strategies, though it took longer to train. This work follows the same general strategy.

3. PROPOSED TECHNIQUES

As alluded to above, this work explores the two questions posed in section 1. Firstly, we consider the case in which an existing, automatically-derived vision system for grading produce is extended to accommodate a new feature. We then move on to examine how new classes of feature may be accommodated.

3.1. Incorporating additional features using mutation

After the vision system is trained on samples of produce to generate a working solution to a particular problem and that solution has been validated to work well on unseen test data, it can be used in the field. However, if additional useful properties of the samples are subsequently found, it may be that the classification accuracy could be improved by incorporating the new properties. Clearly, it will be advantageous if this can be done by adapting the already-trained classifier more quickly than training from scratch.

Following ideas proposed in [5], the authors mutate the existing solution with additional operators that introduce the new features: First, the additional features are selected randomly to form sub-trees. Second, the original classifier are mutated with them to produce solutions. Third, the fitness of the new programs is determined on the training data. If the computed fitness of a solution can reach the goal, the system returns it as a classifier immediately; otherwise, normal GP evolution proceeds. This method is illustrated in figure 1.

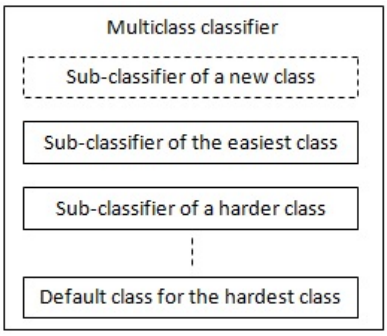

Fig. 2. Adding a new class to a sequence of evolved binary classifiers

3.2. Adding new classes

As described in [1], a trained multiclass vision system comprises a sequence of binary classifiers. Rather than requiring each binary classifier to distinguish one class from all others, their approach is to use the class confusion matrix resulting from the application of k-means to order the classes in terms of difficulty; the binary classifiers are then evolved in terms of increasing difficulty. After each classifier has been obtained, the corresponding training data are removed from the fitness function calculation, which means that the program that has to distinguish the more difficult classes need consider fewer and fewer other classes.

Clearly, if a new class of samples is introduced into any classification system, it needs to be re-trained. For most approaches, this means determining class boundaries or, for GP-trained classifiers, learning new programs *ab initio*. However, for the case described in the previous paragraph in which there is a sequence of binary classifiers, it is necessary only to be able to distinguish the new class from all others, a somewhat simpler task. This idea is shown in figure 2, where the classifier for distinguishing the new class is prepended to the existing sequence.

4. EXPERIMENTS AND DISCUSSIONS

4.1. Introducing new features using mutation

As a starting point, object classifiers were generated based on shape features only. Additional features (from Yimyam and Clark [2]) that describe color and texture properties were then added to the GP 'engine' and mutated with the original classifier as described in the previous section to give a population of new classifiers, which were then evolved. This procedure is demonstrated in five experiments: surface fault inspection on

Table 3. Genetic programming parameters

Parameter	Setting
Population size	500
Generation no.	50
Max tree size	100
Fitness function	$(FP + FN)/N$
Selection method	Tournament ($t = 2$)
Crossover rate	70%
Mutation rate	20%
Reproduction rate	10%
Termination criteria	Fitness = 0
	Maximum generation reached

mandarins, quality grading on purple sticky rice, type classification on forage crops, and maturity determination of tomatoes and limes; each experiment includes ten sub-experiments with different training data. Approximately half the numbers of samples were selected randomly for training, the remaining samples being used for testing. The GP parameters used in all the experiments in this paper are shown in Table 3. The experimental results of the shape classifier mutated to include color and texture features are compared with the original (shape-only features) and the use of all features *ab initio* in Table 1.

Mandarin peel inspection. This task aims to inspect defective skin of mandarins in a cultivar of Sai Nam Pueng. Some 43 images of fruits with good skin color and 54 exhibiting scarring were employed for inspection. The extended classifier was able to achieve 73.7% accuracy, 10% better than the shape-only classifier but around 2% poorer than was obtained using *ab initio* evolution.

Purple sticky rice grading. This experiment intends to classify purple sticky rice qualities. There were 88 paddies and 480 broken, brownish purple and pure purple kernels. As can be seen, the performance of the classifier evolved to use the additional features achieved better than shape-only classification or *ab initio* evolution.

Forage crop type classification. Some types of forage crops are similar in shape, color and texture, so distinguishing them is a real challenge for a vision system. Five types of forage crops (*Plicatulum*, *Paspalum Atratum*, *Cavalcade*, *Sanpatong Vigna* and *Ruzi*) were classified using sample 300 seeds of each type. The average performance of the proposed approach exceeded that of a shape-only classifier, but minimally lower than that of using *ab initio* evolution with all features. Additionally, almost half of the proposed classifiers were obtained directly from the mutation process described above, without having to subject them to further evolution.

Determining the maturity and size of cherry tomatoes. Cherry tomatoes, are green when ripe, changing to orange and then red with time. Sample images were collected in five groups comprising 200 green-colored and large, 80 green-colored and small, 120 orange-colored and large, 110 red-colored and large, and 190 red-colored and (very) small. The mean accuracy results demonstrate the effectiveness of the incorporating the new features, clearly out-performing the shape-only classifiers and being slightly more effective than by performing *ab initio* evolution with the full set of classifiers. Moreover, the proposed mutation process yielded some working classifiers without necessitating evolution.

Lime maturity classification. Ripe limes are pure green in color, subsequently changing to yellow and brown. The sample images were of fruits of roughly the same size but different ripenesses, with 40 images of appropriately-ripe, 40 images of highly-ripe and 44 images of overly-ripe fruit. It can be seen that the mean accuracy of the proposed technique is significantly higher than that of the shape-only classifier but slightly lower than can be achieved using *ab initio* evolution. Most of the classifiers generated by the proposed mutation technique needed no further evolution.

4.2. Adding a new class of samples

To explore the addition of a new class of samples, a binary classifier was first evolved to distinguish two classes; then, a new class was added with its own binary classifier — and so on, until the last new class was introduced. The binary classifiers were then assembled to form the final multiclass classifier. Five experiments were performed on the purple sticky rice, forage crop, tomato and lime datasets employed above, plus a further experiment in regard to mandarin sorting. The accuracy results of the proposed approach are shown in Table 2 and are compared with the approach of evolving classifiers *ab initio* [1].

Mandarin sorting. This experiment aims to classify the size of mandarins. The samples were divided into four size groups: 28 giant-sized, 32 big, 32 medium, and 44 small. There were four groups of samples so a new class was added two times. The mean accuracy obtained is over 3% better than by evolving a solution *ab initio*.

Purple sticky rice grading. The samples were divided into four groups. A binary classifier was generated for two groups of the samples, and the two other classes were introduced one by one. The modified program accomplished 96.3% accuracy, nearly equal to the accuracy achieved by the *ab initio* evolution.

Forage crop type classification. Five types of crop were available, so a new class was introduced three times to generate the final multiclass classifier. As can be seen, the proposed technique improved accuracy by about 6%.

Maturity and size of cherry tomatoes. There were five groups of samples, so a new class was introduced three times. The proposed approach achieves about 5% better accuracy than *ab initio* evolution.

Lime maturity classification. As the sample limes comprised of three varieties, only one additional class was incor-

Table 1. Classification results regarding adding features

Technique	Mandarin Peel			Purple sticky rice			Forage crops			Tomatoes			Limes		
	Avg.	s.d.	Best	Avg.	s.d.	Best	Avg.	s.d.	Best	Avg.	s.d.	Best	Avg.	s.d.	Best
Shape only	63.7	6.38	75.0	97.5	1.06	99.48	76.4	4.63	80.1	59.8	3.07	64.2	55.6	6.05	67.7
Ab initio evolution	75.4	5.54	81.3	98.4	0.90	100.0	91.0	1.60	94.1	90.5	4.25	94.2	90.2	5.45	96.8
Proposed method	73.7	4.93	79.2	98.6	0.88	100.0	88.7	2.58	91.1	91.4	3.11	94.5	87.9	4.25	93.6

Table 2. Classification results regarding adding new classes

Technique	Mandarins			Purple sticky rice			Forage crops			Tomatoes			Limes		
	Avg.	s.d.	Best	Avg.	s.d.	Best	Avg.	s.d.	Best	Avg.	s.d.	Best	Avg.	s.d.	Best
Ab initio evolution	83.9	4.22	89.7	97.9	0.83	99.1	83.6	2.64	87.2	88.7	3.15	93.9	89.2	5.48	96.8
Proposed method	87.1	2.28	91.2	96.3	0.39	97.0	89.4	3.87	94.1	93.8	1.45	95.8	92.6	3.15	95.2

porated. As shown in Table 2, the average accuracy of the proposed approach is again a little better than *ab initio* evolution.

5. CONCLUSIONS AND FUTURE WORK

This paper has demonstrated that a computer vision system initially evolved to solve a specific multiclass problem can be adapted to accommodate new features or further classes. For new features, the approach taken is to 'mutate in' the feature into the population of programs and then, if necessary, evolve them. Experiments on a number of tasks taken from the general area of produce grading demonstrated that this approach often yields more accurate complete systems than *ab initio* evolution using all the features. Moreover, the proposed method is considerably faster than *ab initio* evolution. In roughly half of the cases considered, the mutation process was able to deliver programs that performed well enough on the training data that subsequent evolution was unnecessary; however, this is likely to be a consequence of the advantage of employing color and texture relative to shape alone — we do not expect this to be achievable in more general vision tasks.

Adding new classes of feature was achieved by evolving a single binary classifier, executed at the beginning of the sequence of binary classifiers to yield overall multiclass classification. This appears to work reasonably well, out-performing the approach of [1] on a number of occasions. However, the authors are of the opinion that the approach could be improved by examining the relationship between class confusion and classification accuracy. Further work is in hand to explore this.

6. REFERENCES

[1] O. Oechsle and A. F. Clark, "Feature extraction and classification by genetic programming," in *International Conference on Vision Systems (ICVS)*, 2008.

[2] P. Yimyam and A.F. Clark, "Agricultural produce grading by computer vision using genetic programming," in *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference on*, 2012, pp. 458–463.

[3] O. Oechsle, *Towards the Automatic Construction of Machine Vision Systems using Genetic Programming*, Ph.D. thesis, Computer Science and Electronic Engineering, 2009.

[4] R. K. Rao, K. Tun, and S. Lakshminarayanan, "Genetic programming based variable interaction models for classification of process and biological systems," *Industrial & Engineering Chemistry Research*, vol. 48, no. 10, pp. 4899–4907, 2009.

[5] L. Zhang and A. K. Nandi, "Fault classification using genetic programming," *Mechanical Systems and Signal Processing*, vol. 21, no. 3, pp. 1273–1284, 2007.

[6] M. Zhang and M. Lett, "Genetic programming for object detection: Improving fitness functions and optimising training data," *The IEEE Intelligent Informatics Bulletin*, vol. 7, no. 1, pp. 12–21, Dec. 2006.

[7] J. Li, X. Li, and X. Yao, "Cost-sensitive classification with genetic programming," in *IEEE Congress on Evolutionary Computation*, 2005, vol. 3, pp. 2114–2121 Vol. 3.

[8] J. Hong and S. Cho, "Lymphoma cancer classification using genetic programming with snr features," in *in Proccedings of 7th European Conference, EuroGP 2004*, 2004, pp. 78–88.

[9] S. Hengpraprohm and P. Chongstitvatana, "A genetic programming ensemble approach to cancer microarray data classification," in *Innovative Computing Information and Control, 2008. ICICIC '08. 3rd International Conference on*, 2008, pp. 340–340.

[10] H. Guo and A.K. Nandi, "Breast cancer diagnosis using genetic programming generated feature," in *Machine Learning for Signal Processing, 2005 IEEE Workshop on*, 2005, pp. 215–220.

[11] T. Loveard and V. Ciesielski, "Representing classification problems in genetic programming," *Evolutionary Computation*, vol. 2, pp. 1070–1077, 2001.