Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition Letters 25 (2004) 1519–1531
Pattern Recognition Letters
www.elsevier.com/locate/patrec

Synthesizing feature agents using evolutionary computation

Bir Bhanu *, Yingqiang Lin

Center for Research in Intelligent Systems, University of California, Riverside, CA 92521, USA

Abstract

In this paper, genetic programming (GP) with smart crossover and smart mutation is proposed to discover integrated feature agents that are evolved from combinations of primitive image processing operations to extract regions-of-interest (ROIs) in remotely sensed images. The motivation for using genetic programming is to overcome the limitations of human experts, since GP attempts many unconventional ways of combination, in some cases, these unconventional combinations yield exceptionally good results. Smart crossover and smart mutation identify and keep the effective components of integrated operators called "agents" and significantly improve the efficiency of GP. Our experimental results show that compared to normal GP, our GP algorithm with smart crossover and smart mutation can find good agents more quickly during training to effectively extract the regions-of-interest and the learned agents can be applied to extract ROIs in other similar images.

Keywords: Feature synthesis; Genetic programming; ROI extraction; Smart crossover and smart mutation

1. Introduction

Image segmentation and labeling of terrain regions is an important task in remote sensing application (Mvogo et al., 2000; Jeon et al., 2002; Katartzis et al., 2001; Segl and Kaufmann, 2001; Acqua and Gamba, 2001; Dong et al., 2001). The quality of this task is primarily dependent on the data and features used as input. There are many kinds of features that can be used, and the question is what are the appropriate features or how to synthesize features, particularly useful for segmentation/labeling, from the primitive features extracted from images. Usually, there are almost infinite ways of combining primitive features to form synthesized composite ones. It is the human image experts who, relying on their rich experience, figure out a smart way of combination. The task of finding a good combination is equivalent to finding a good point in the search space of agents formed by the combination of primitive operations (also called primitive operators) on images.

However, limited by their speed, previous experience or even bias, the human experts can only try a very limited number of conventional combinations. Genetic programming (GP), on the other hand, may try many unconventional combinations that may yield exceptionally good

* Corresponding author. Tel.: +1-909-787-3954; fax: +1-909-787-3188.
E-mail addresses: bhanu@cris.ucr.edu (B. Bhanu), yqlin@cris.ucr.edu (Y. Lin).

0167-8655/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2004.06.005

results. Also, genetic programming can explore a much larger portion of agent space due to the inherent parallelism of GP and the speed of computer. The search performed by GP is guided by the goodness of agents in the population. As the search proceeds, GP will gradually shift the population to the portion of the space that contains good agents. The crossover operation is the major mechanism used by GP to search the agent space and the mutation operation is employed to increase the diversity of the population to avoid the premature convergence. However, in the traditional crossover and mutation, their locations are randomly selected, leading to disrupting the effective components (subtree in this paper) within agents and thus greatly reducing the efficiency of GP. It is very important for GP to identify and keep those effective components.

In this paper, we use genetic programming to generate feature agents for terrain labeling. The individuals in our GP-based learning are agents represented by binary trees whose internal nodes represent the pre-specified primitive operators and the leaf nodes represent the original image or the primitive feature images generated from the original image. The primitive feature images are pre-determined, and they are not the output of the pre-specified primitive operators. After applying an agent on the original image or the primitive feature images, the output image of the agent is segmented to yield a binary image or mask. The binary mask is used to extract the particular kind of terrain regions from the original image. To improve the efficiency of GP, we propose smart crossover and smart mutation to identify and keep the effective components of an agent.

2. Motivation, related research and contributions

2.1. Motivation for intelligent GP operators

Search operators are one of the most important parts of any machine learning system. They define the manner in which the learning system moves through the space of candidate solutions. In this paper, crossover and mutation are major search operators used by GP to search the huge agent space. Finding good agents is a tough task and it is very important to design intelligent crossover and mutation operators in order to improve the efficiency of GP.

As it is well known, crossover is the predominant mechanism employed by GP to perform search. The power of crossover lies in the fact that by swapping subtrees between two good agents (parents), the good components (subtrees) in these two parent agents can be assembled together into offspring agents (children) and the offspring agents may be better than both parents. However, although crossover can assemble good components to yield better agents, it is also a destructive force in the sense that it can disrupt good agents due to the random selection of crossover points. When the search begins, since the initial population is randomly generated, it is unlikely that an agent contains good, especially large good, components. The probability of crossover breaking up a good component is small. At this time, crossover is a constructive force and the average fitness of the agent population is increased. Small good components are generated and assembled into larger and larger good components as search proceeds. As more and more agents contain large good components to achieve high fitness, the good component accounts for a large portion of a whole agent and the agent becomes more and more fragile because the good components are more prone to being broken up by subsequent crossover due to the random selection of crossover point. The crossover can damage the fitness of an agent in ways other than disrupting good components. Usually, a good component is moved into an inhospitable context, that is, the crossover inserts a good component into an agent that does not use the good component in any useful way or other nodes in the agent cancel out the effect of the good component. According to Nordin and Banzhaf (1995), crossover has an overwhelmingly negative effect on the fitness of the offspring of the crossover, especially in the later stage of GP runs. One of the most serious problems of GP is the convergence of a population. According to Banzhaf et al. (1998), what has not been achieved within 50 generations will never be achieved. A serious weakness of evolutionary algorithm is that the

population recombined repeatedly will develop uniformity sooner or later. Diversity is the key to finding remedy to this situation, so mutation is introduced to maintain the diversity of the agent population to prevent premature convergence. However, in the later stages of GP runs when more and more agents contain large good components, the random selection of mutation point leads to the high probability of disrupting good components and makes mutation a destructive force. When both crossover and mutation become negative factors in the GP search, it is very unlikely that a better agent will be generated. To improve the efficiency and effectiveness of GP, it is highly desired that good components can be identified and kept from destructive crossover and mutation operators.

2.2. Related research

Genetic programming, an extension of genetic algorithm, is first proposed by Koza (1994) and it has been used in image processing, object detection, object recognition, etc. Poli (1996) uses GP to develop effective image filters to enhance and detect features of interest or to build pixel-classification-based segmentation algorithms. Bhanu and Lin (2002) use GP to generate agents for ROI extraction. Their experimental results show that GP is a viable way to search the agent space. They also find that random selection of crossover and mutation points may degrade the performance of GP. Stanhope and Daida (1998) use GP paradigms for the generation of rules for target/clutter classification and rules for identification of objects. To perform these tasks, previously defined feature sets are generated on various images and GP is used to select relevant features and methods for analyzing these features. Howard et al. (1999) apply GP to automatic detection of ships in low-resolution SAR (synthetic aperture radar) imagery using an approach that evolves detectors. Roberts and Howard (1999) use GP to develop automatic object detectors in infrared images.

To improve the efficiency of GP, Tackett (1994) devises a method called brood recombination to reduce the destructive effect of crossover. In this method, when crossover is performed, many offspring are generated from two parents and only the best two offspring are kept. D'haeseleer (1994) devises strong context preserving crossover (SCPC) to preserve the context. SCPC only permits crossover between nodes that occupy exactly the same position in the two parents. He finds modest improvement in results by mixing regular crossover and SCPC. Smith (1996) proposes a conjugation operator for GP to transfer genetic information from one individual to another. In his conjugation method, the parent with higher fitness becomes the donor and the other with lower fitness becomes the recipient. The conjugation operator is different from crossover and it is proposed to simulate one of the ways in which individuals exchange genetic materials in nature. Ito et al. (1998) propose a depth-dependent crossover for GP in which the selection is varied according to the depth of a node. A node closer to the root node of the tree has a better chance of being selected as a crossover point to lower the chance of disrupting small good components near leaves. Their experimental results show the superiority of the depth-dependent crossover to the random crossover in which crossover points are randomly selected.

Unlike the work of Stanhope and Daida (1998) and Howard et al. (1999), the input and output of each node of a tree in our system are images, not real numbers. Also unlike the work of Ito et al. (1998) that uses only the syntax of a tree (the depth of a node), the smart crossover and smart mutation proposed in this paper evaluate the performance at each node to determine the interactions among them and use the fitness value at each node to determine the crossover and mutation points.

In many cases, the purpose of ROI extraction in remotely sensed images is to detect particular kinds of terrain regions or objects. Jeon et al. (2002) propose a GA-based road detection algorithm in spaceborne SAR images. GA is applied to find an optimal grouping method to group curve segments representing the candidate positions of roads for accurate road detection. Katartzis et al. (2001) describe a model-based technique combining both local and global criteria to extract roads in airborne SAR images. Its main advantage is the good detection performance in heavily textured

environments and its ability to identify elongated structures independent of their size. Its main limitation is that it is not unsupervised due to the setting of parameters. Segl and Kaufmann (2001) combine supervised shape classification with unsupervised image segmentation in an iterative procedure for the detection of small objects from high spatial resolution satellite images. Their technique facilitates monitoring of urban area expansions and the distribution of vegetation patterns. Acqua and Gamba (2001) present a fuzzy approach to the analysis of SAR images of urban environments. They focus on the street tracking and extraction. The clustering is applied to group initially segmented urban classes such as roads and built areas to extract structures of interest. Dong et al. (2001) propose an approach to the classification of SAR images. The area analysis-based approach is more accurate than a pixel-based approach, since a uniform area provides reliable measurement statistics and texture characteristics. As compared to the above approaches, this paper is an attempt to develop a GP-based learning approach to automatically generate effective new features based on the available primitive ones for the detection of terrain regions or objects in remotely sensed images.

2.3. Contributions of this paper

The paper presents a novel approach based on genetic programming for synthesizing complex feature extractors from primitive image processing operators and features. To improve the effectiveness and efficiency of GP, the smart crossover and smart mutation operators are proposed to identify and keep good components of agents. The identification of good components is based on the performance of each internal node and the interactions among them. Identifying and keeping the good components make it more likely for GP to assemble them together to produce better agents quickly. The primitive operators and primitive features in our system are very basic and domain-independent, not specific to a kind of imagery, leading to many applications for our GP-based learning system to a wide variety of images.

3. Technical approach

In our GP-based approach, individuals are agents represented by binary trees. The search space of GP is the space of all possible feature agents. The space is very large. To illustrate this, consider only a special kind of binary tree, where each tree has exactly 30 internal nodes and one leaf node and each internal node has only one child. For 17 primitive operators and only one primitive feature image, the total number of such trees is 17^{30}. It is extremely difficult to find good agents from this vast space unless one has a smart search strategy.

3.1. Design considerations

There are five major design considerations, which involve determining the set of terminals, the set of primitive operators, the fitness measure, the parameters for controlling the run, and the criterion for terminating a run. (1) *The set of terminals*—the set of terminals used in this paper are seven primitive feature images generated from the original image: the first one is the original image; the others are mean and standard deviation images obtained by applying templates of sizes 3×3, 5×5 and 7×7. These images are the input to the agents. GP determines which operations are applied on them and how to combine the results. (2) *The set of primitive operators*—a primitive operator takes one or two images as input image(s), performs some primitive operations on them and stores the result in a resultant image. Currently, 17 primitive operators are used by GP to compose agents and they are shown in Fig. 1. (3) *The fitness measure*—the fitness value of an agent is computed in the following way. Suppose during training G and G' are foregrounds in the ground-truth image and the resultant image of an agent respectively. Let $n(X)$ denote the number of pixels within region X, then: $\text{Fitness} = n(G \cap G')/n(G \cup G')$. The fitness value is between 0 and 1. If G and G' are completely separated, the value is 0; if G and G' are completely overlapped, the value is 1. (4) *Major parameters controlling the run*—the key parameters are the population size M, the number of generations N, the crossover rate and the mutation rate. (5) *Ter-*

1. ADD_OP: A + B. Add two images.
2. SUB_OP: A – B. Subtract image B from image A.
3. MUL_OP: A * B. Multiply images A and B.
4. DIV_OP: A / B. Divide image A by image B (If the pixel in B has value 0, the corresponding pixel in the resultant image takes the maximum pixel value in A).
5. MAX2_OP: A max B. Each pixel in the resultant image takes the larger value of the corresponding pixels in images A and B.
6. MIN2_OP: A min B. Each pixel in the resultant image takes the smaller value of the corresponding pixels in images A and B.
7. ADD_CONST_OP: A + c. Increase pixel value by c.
8. SUB_CONST_OP: A - c. Decrease pixel value by c.
9. MUL_CONST_OP: A * c. Multiply pixel value by c.
10. DIV_CONST_OP: A / c. Divide pixel value by c.
11. SQRT_OP: sqrt(A). For each pixel p with value v, if $v \geq 0$, change its value to $\sqrt{v}$. Otherwise, to $-\sqrt{-v}$.
12. LOG_OP: log(A). For each pixel p with value v. if $v \geq 0$, change its value to log(v). Otherwise, to –log(-v).
13. MAX_OP: max(A). Replace the pixel value by the maximum pixel value in a 3×3, 5×5 or 7×7 neighborhood.
14. MIN_OP: min(A). Replace the pixel value by the minimum pixel value in a 3×3, 5×5 or 7×7 neighborhood.
15. MED_OP: med(A). Replace the pixel value by the median pixel value in a 3×3, 5×5 or 7×7 neighborhood.
16. REVERSE_OP: rev(A). Reverse the pixel value. Suppose the maximum and minimum pixel values of image A are *Vmax* and *Vmin* respectively. If a pixel has value v, change its value to $Vmax - v + Vmin$.
17. STDV_OP: stdv(A). Obtain standard deviation image of image A by applying a template of size 3×3, 5×5 or 7×7.

Fig. 1. Seventeen primitive operators (A and B are images of the same size and c is a constant).

mination—The GP stops whenever it finishes the pre-specified number of generations or whenever the best agent in the population has fitness value greater than the fitness threshold.

3.2. Reproduction, crossover and mutation

The GP searches through the agent space to find good agents. By searching through the agent space, GP gradually adapts the population of agents from generation to generation and improves the overall fitness of the whole population. The search is done by reproduction, crossover and mutation operations. The initial population is randomly generated and the fitness of each agent is evaluated. The reproduction operation involves selecting an agent from the current population. In this research, we use tournament selection, where a number of agents are randomly selected from the current population and the one with the highest fitness value is copied into the new population.

3.2.1. Crossover

Traditionally, to perform crossover, two agents are selected on the basis of their fitness values. The higher the fitness, the more likely the agent will be selected for crossover. These two agents are called parents. One internal node in each of these two parents is randomly selected, and the two subtrees with these two nodes as root are exchanged between the parents. In this way, two new agents, called offspring, are created. The two internal nodes selected during crossover are called crossover point. Since crossover point is selected randomly, the traditional crossover is also called random crossover. Usually, an agent with high fitness value contains a component (subtree) that is efficient in extracting the region of interest. However, due to the random selection of crossover point, random crossover may select an internal node within this component and disrupt it, leading to the great reduction in the efficiency of genetic programming.

In order to avoid the above problem, we propose smart crossover that identifies and keeps the good components of an agent. In the smart crossover, the output image of each node, not just the resultant image from the root node, is evaluated and the fitness value of each node is recorded. Based on the fitness values of all the nodes, the edges of an agent are classified as good edge or bad edge. If the fitness value of a node is smaller than that of its parent node, the edge linking these two nodes is a good edge. Otherwise, the edge is

labeled as bad edge. During crossover, all the bad edges are identified and one of them is randomly selected. The child node of the selected bad edge is chosen as the crossover point. If an agent has no bad edge, the crossover point is selected randomly. We use ε-greedy policy to determine which crossover is used. With probability ε, the smart crossover is invoked; with probability $1-\varepsilon$, the random crossover is invoked. In this paper, ε is set to 0.9.

3.2.2. Mutation

In order to avoid premature convergence, mutation is introduced to randomly change the structure of some agents to help maintain the diversity of the population. The agent selected for mutation is selected randomly. Also, both random mutation and smart mutation are used. In the random mutation, a node of an agent is randomly selected as mutation point, and the subtree rooted at this node is replaced by another randomly generated tree. The resulting new agent replaces the old one in the population. In the smart mutation, one of the bad edges is randomly chosen, and the parent node of the bad edge is selected as mutation point. If the parent node has only one child, the parent node is deleted and the child node is directly linked to the grand parent node (the parent node of the deleted parent node); if the parent node has two children, the parent node and the subtree rooted at the child node with smaller fitness value are deleted and the other child node is directly linked to the grand parent node. If the grand parent node does not exist, the child becomes the root node. The smart mutation attempts to delete internal node that decreases the fitness value and keeps the good component. If the agent has no bad edge, the mutation point is selected randomly. Similar to the crossover case, ε-greedy policy determines which mutation is invoked and the value of ε is 0.9 in this paper.

3.3. Steady-state and generational genetic programming

In steady-state GP, two agents are selected based on their fitness values for crossover. The children of this crossover, perhaps mutated, replace a pair of agents with the smallest fitness values. The two children are executed and evaluated immediately. Then another two agents are selected for crossover. This process is repeated until crossover rate is satisfied. In generational GP, two agents are selected based on their fitness values for crossover and the two offspring of the crossover are kept. At this time, the two offspring are not put into the current population and they will not participate in the following crossover operations on the current population. The above process is repeated until crossover rate is satisfied. After crossover operations are finished, all the children resulting from the crossover operations are combined with the current population and the tournament selection is employed to select agents from the combined population to generate the new population of next generation. For both GP algorithms, the mutation is performed after crossover in each generation and we adopt an elitism replacement method to keep the best agent from generation to generation.

4. Experiments

Various experiments were performed to test the efficacy of genetic programming in extracting regions of interest from real synthetic aperture radar (SAR) images. We show three selected examples. In each of three examples, GP is applied to the training images to generate an agent for a particular kind of terrain region and the agent is then applied to the testing images. It is worth noting that the training and testing images are different images and the ground truth is used only during training. In all the experiments, the maximum size (the number of internal nodes) of an agent is 30 and the threshold value for segmentation is 0. In each experiment, both normal genetic programming (GP with random crossover and random mutation) and smart genetic programming (GP with smart crossover and smart mutation) are applied to the training image. For the purpose of objective comparison, we invoke the normal GP and smart GP with the same set of parameters in each experiment. The population size is 100, the number of generations is 100, the fitness threshold value is 0.95, the crossover rate is 0.6, and the mutation rate is 0.1.

4.1. Example 1—Extracting river from real SAR images

We have two SAR images containing river and they are shown in Fig. 2. Fig. 2(a) is used in training and GP generates an agent to extract the river in the image. The first primitive feature image is the original image and the mean and standard deviation images are shown in Fig. 3, where 3×3, 5×5 and 7×7 are the sizes of the templates used to obtain the primitive feature images from the original image. The generated agent is applied to the SAR image shown in Fig. 2(d) to test its efficacy in extracting the river. The steady-state genetic programming is used to generate the agent. Fig. 2(b) shows the regions of interest extracted by the best agent generated by normal GP. The fitness value of the best agent is 0.74 and it is not very good. Two reasons account for this. *First*, the river in Fig. 2(a) accounts for a small percentage of the total area in the image. *Second*, there are some islands in the river. These islands are similar to the field, but they are not excluded from the ground truth. Fig. 2(c) shows the regions of interest extracted by the best agent generated by smart GP. The fitness values of the best agent is 0.77 and it is shown in Fig. 4, where PM_IM0 is original image, and PF_IM1, PF_IM2 and PF_IM3 are the primitive feature images as shown in Fig. 3. The agent has 30 internal nodes and 12 of them contain MED_OP primitive operator, which is very useful in speckle noise reduction. It is possible to have a more compact tree representation of this agent.

We apply the best agents from normal and smart GP to the image shown in Fig. 2(d). Fig. 2(e) shows the region of interest extracted by the best agent from normal GP. The fitness value of the extracted region is 0.84. Fig. 2(f) shows the region

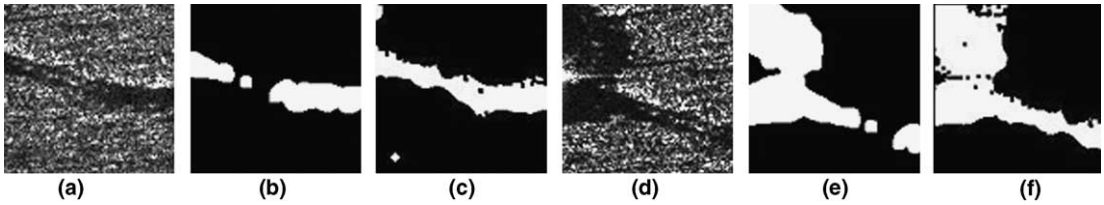

Fig. 2. SAR images containing river and field: (a) river vs. field (training); (b) best ROI in training (normal GP); (c) best ROI in training (smart GP); (d) river vs. field (testing); (e) best ROI in testing (normal GP); (f) best ROI in testing (smart GP).

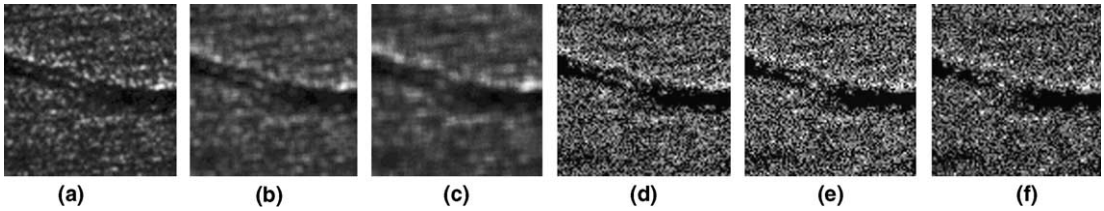

Fig. 3. Primitive feature images for training SAR image containing river and field in Fig. 2: (a) 3×3 mean image (PF_IM1); (b) 5×5 mean image (PF_IM3); (c) 7×7 mean image (PF_IM5), (d) 3×3 stdv image (PF_IM2); (e) 5×5 stdv image (PF_IM4); (f) 7×7 stdv image (PF_IM6).

```
(MIN_OP (DIV_OP (LOG_OP (MED_OP (MED_OP (MED_OP (MED_OP (MED_OP (MED_OP (MED_OP
(MED_OP (MED_OP (DIV_OP (STDV_OP (MIN2_OP (PF_IM1 PF_IM2) (STDV_OP (STDV_OP
(ADD_CONST_OP (SUB_OP (MED_OP PF_IM2) (DIV_CONST_OP PF_IM0)))))))))))))) (SUB_OP (SUB_OP
(MED_OP PF_IM2) (DIV_CONST_OP PF_IM0)) (MIN2_OP PF_IM1 (STDV_OP (MIN2_OP PF_IM1 (SUB_OP
(MED_OP PF_IM2) PF_IM3)))))))
```

Fig. 4. River agent in LISP notation learned by smart GP.

Table 1
Confusion matrices for training and testing images containing river in Example 1

Image—pixels	River	Background
River—2192	1960 (0.89)	232 (0.11)
River—4805	4506 (0.94)	299 (0.06)
Background—14,192	365 (0.03)	13,827 (0.97)
Background—11,579	682 (0.06)	10,897 (0.94)

of interest extracted by the best agent from smart GP. The fitness value of the extracted region is 0.82. Table 1 shows the confusion matrices when the agent obtained by smart GP during the training is applied to both training and testing images. It shows the number (percentage) of river and background pixels that are correctly and incorrectly detected.

4.2. Example 2—Extracting field from real SAR image

Fig. 5 shows two SAR images containing field and grass. Fig. 5(a) is used in training and Fig. 5(d) is used in testing. We consider extracting field from a SAR image containing field and grass as the most difficult task among the three experiments, since the grass and field are similar to each other. The generational genetic programming is used to generate the agent. Fig. 5(b) shows the region extracted by the best agent found by normal GP and Fig. 5(c) shows the region extracted by the best agent found by smart GP. The fitness values of the best agents from normal GP and smart GP during training are 0.87 and 0.90 respectively. We apply the best agents from normal and smart GPs to the image in Fig. 5(d). Fig. 5(e) shows the region of interest extracted by the best agent from normal GP, and the fitness value of the extracted region is 0.68. Fig. 5(f) shows the region of interest extracted by the best agent from smart GP, and the fitness value of the agent is 0.83. The best agent evolved by smart GP is much better than that evolved by normal GP. Table 2 shows the confusion matrices obtained by using this agent from smart GP on the training and testing images.

4.3. Example 3—Extracting lake from real SAR images

Two SAR images contain lake. The first one contains a lake and field, and the second one contains a lake and grass. Fig. 6(a) shows the original image containing lake and field and Fig. 6(d) shows the image containing lake and grass. We use the SAR image containing the lake and field as the training image and apply the agent generated by GP to the SAR image containing the lake and grass. The steady-state genetic programming is used to generate the agent. Fig. 6(b) shows the region extracted by the best agent from normal GP and Fig. 6(c) shows the region extracted by the best agent from smart GP. Their fitness values are 0.93 and 0.94 respectively. We apply the best agents from normal and smart GPs to the image in Fig. 6(d). Fig. 6(e) shows the region of interest

Table 2
Confusion matrices for training and testing images containing field in Example 2

Image—pixels	Field	Background
Field—9696	9509 (0.98)	187 (0.02)
Field—8215	7525 (0.92)	690 (0.08)
Background—6688	906 (0.14)	5782 (0.86)
Background—8169	897 (0.11)	7272 (0.89)

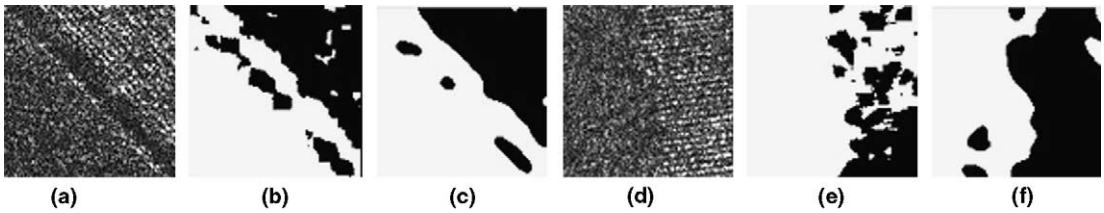

Fig. 5. SAR images containing field and grass: (a) field vs. grass (training); (b) best ROI in training (normal GP); (c) best ROI in training (smart GP), (d) field vs. grass (testing); (e) best ROI in testing (normal GP); (f) best ROI in testing (smart GP).

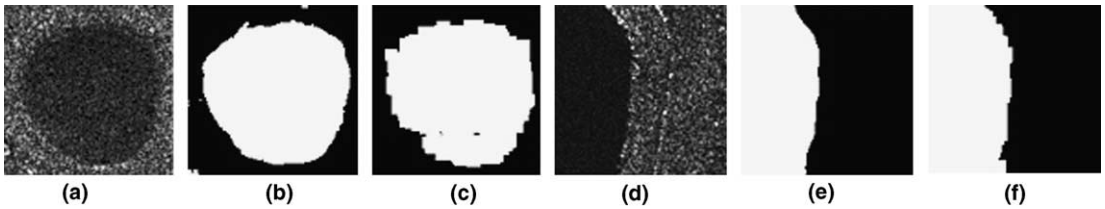

Fig. 6. SAR image containing lake and field: (a) lake vs. field (training); (b) best ROI in training (normal GP); (c) best ROI in training (smart GP); (d) lake vs. grass (testing); (e) best ROI in testing (normal GP); (f) best ROI in testing (smart GP).

extracted by the best agent from normal GP and its fitness value is 0.92. Fig. 6(f) shows the region of interest extracted by the best agent from smart GP and its fitness value is 0.97, which is higher than that from normal GP. Table 3 shows the confusion matrices obtained by using this agent from smart GP on the training and testing images.

We also apply the agent learned by smart GP to a larger SAR image containing a lake, which is shown in Fig. 7(a). The ground-truth and the region extracted are shown in Fig. 7(b) and (c) respectively. The fitness of the extracted region with respect to the ground-truth is 0.91. The high fitness value indicates that in order to obtain an agent, it is not necessary to apply GP to a whole training image. GP can be applied to a carefully chosen region (called training region) from the training image. The generated agent can then be applied to the large size training/testing images. Performing training on a small region can greatly reduce the training time, thus, making it practical for the GP system to be embedded in other learning systems that can improve the efficiency of GP by adapting GP system parameters such as crossover and mutation rates based on its performance.

It is worth noting that since elitism mechanism is adopted, the fitness value of the best agent dose not decrease from generation to generation. However, the average fitness value of the population may slightly decrease, although the fluctuation is very small. The general trend is that the average fitness value increases quickly within the first 20–30 generations. Then the marginal improvement is achieved in the rest of the generations.

Table 3
Confusion matrices for training and testing images containing lake in Example 3

Image—pixels	Lake	Background
Lake—9698	9368 (0.97)	330 (0.03)
Lake—7108	6947 (0.98)	161 (0.02)
Background—6686	301 (0.05)	6385 (0.95)
Background—9276	51 (0.01)	9225 (0.99)

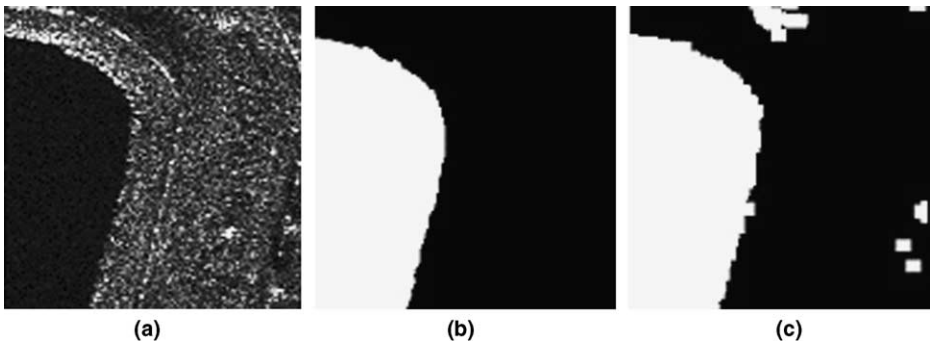

Fig. 7. Another SAR image containing lake and field: (a) lake vs. field; (b) ground-truth; (c) ROI extracted (smart GP).

4.4. Comparison of normal and smart GPs

We run both normal GP and smart GP on the training SAR images to generate agents for river, field and lake. The experimental results and the comparison between the normal GP and smart GP are shown in Table 4 and Figs. 8 and 9. In Table 4, the "Best fitness" column and the "Average fitness" column show the fitness of the best agent and the population fitness (the average fitness of all the agents in the population) after some generations. The numbers in the parenthesis in the "Best fitness" columns are the fitness values of the best agents on the testing SAR images. Fig. 8 shows how the fitness of the best agent improves as the normal GP and smart GP run. Similarly, Fig. 9 shows the improvement of the population fitness as normal GP and smart GP run. Both Figs. 8 and 9 show that smart GP finds good agents more quickly. Since population fitness is the average fitness values of 100 agents in the population, Fig. 9 demonstrates convincingly that the smart GP is more efficient.

Table 4
The performance of normal GP and smart GP on various SAR images during training (and testing)

River vs. field			Field vs. grass			Lake vs. field		
Generation	Best fitness	Average fitness	Generation	Best fitness	Average fitness	Generation	Best fitness	Average fitness
Normal genetic programming								
0	0.43	0.21	0	0.62	0.44	0	0.65	0.42
48	0.74 (0.84)	0.68	50	0.87 (0.68)	0.86	45	0.93 (0.92)	0.92
Smart genetic programming								
0	0.35	0.11	0	0.62	0.47	0	0.69	0.49
21	0.74	0.69	5	0.85	0.60	5	0.91	0.85
42	0.75	0.70	32	0.89	0.86	17	0.93	0.89
49	0.77 (0.82)	0.74	94	0.90 (0.83)	0.89	26	0.94 (0.97)	0.92

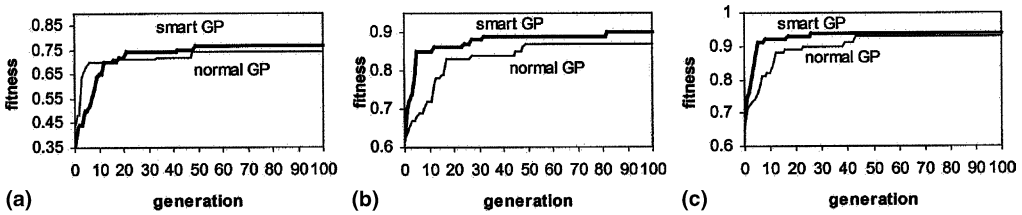

Fig. 8. The fitness of the best agent versus generation: (a) river vs. field; (b) field vs. grass; (c) lake vs. field.

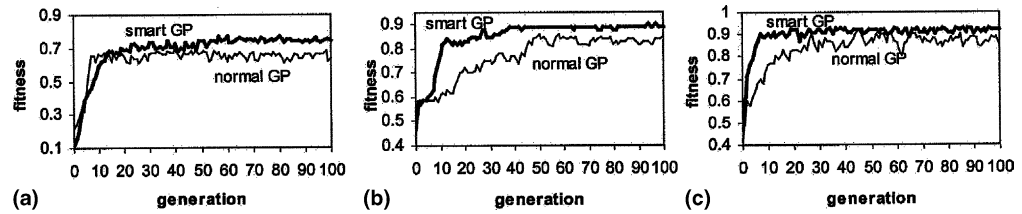

Fig. 9. The fitness of agent population versus generation: (a) river vs. field; (b) field vs. grass; (c) lake vs. field.

4.5. Comparison with traditional ROI extraction algorithms

In order to show the effectiveness of agents learned by GP in ROI (terrain region) extraction, they are compared with two traditional ROI extraction algorithms. Both traditional ROI extraction algorithms use a threshold value to segment an image into foreground and background. The regions consisting of pixels with value greater than the threshold value are called bright regions and their complements are called dark regions. The ROIs may be bright regions or dark regions. If the bright regions have a higher fitness than the dark regions, the bright regions are the ROIs to be extracted. Otherwise, the dark regions are the ROIs to be extracted. The threshold value plays a vital role in the ROI extraction and the difference between these two algorithms lies in the way of selecting an appropriate threshold value.

The *first* traditional ROI extraction algorithm evaluates the performance of every possible threshold value and outputs the ROI extracted by the best one. The performance of learned agents is compared with the performance corresponding to the best threshold value. The performance is the fitness value as defined in Section 3.1.

The Traditional ROI Extraction Algorithm

1. *find the maximum and minimum pixel values of the image.*
2. *if the maximum pixel value is greater than 1000 then*
3. *normalize the pixel values into the range of 0 to 1000. The pixel values are changed according to the following equation.*

$$new\_pixval = (org\_pixval - min\_pixval) / (max\_pixval - min\_pixval) * 1000$$

where new_pixval and org_pixval are the new and original pixel values, respectively and min_pixval and max_pixval are the minimum and maximum pixel values in the original image.

After normalization, the minimum and maximum pixel values are 0 and 1000, respectively.

else

do not normalize the image.

endif

4. *each pixel value between the minimum and maximum pixel values is used as the threshold value and its performance in ROI extraction is recorded.*
5. *select the best threshold value and output its corresponding ROI.*

Fig. 10 shows the ROIs extracted by the first traditional ROI extraction algorithm corresponding to the best threshold value. The fitness values of the extracted ROIs and their corresponding threshold values are also shown.

To extract ROIs from SAR images, the original SAR images are used by the traditional ROI extraction algorithm. Comparing Fig. 10 with the results shown in Sections 4.1–4.3, it can be seen that the agents learned by normal GP and smart GP are more effective in ROI extraction. Actually, its performance is better than the best performance of the traditional ROI extraction algorithm. Furthermore, it takes the traditional ROI extraction algorithm much longer time than that of GP generated agents in ROI extraction, since the traditional algorithm has to try every possible threshold value in order to produce the best

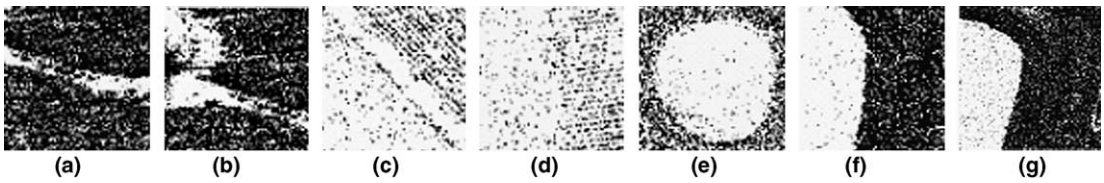

Fig. 10. ROI extracted (white pixels) by the traditional ROI extraction algorithm: (a) river vs. field (fitness = 0.31, threshold = 34); (b) river vs. field (fitness = 0.55, threshold = 38); (c) field vs. grass (fitness = 0.63, threshold = 112); (d) field vs. grass (fitness = 0.53, threshold = 128); (e) lake vs. field (fitness = 0.72, threshold = 36); (f) lake vs. grass (fitness = 0.81, threshold = 24); (g) lake vs. field (fitness = 0.75, threshold = 21).

performance. Thus, the GP generated agents are more efficient.

The *second* traditional ROI extraction algorithm assumes that the pixels within ROI to be extracted have value μ_1 and background pixels have value μ_2, both contaminated with additive Gaussian noise. Thus, the mixture probability density function of a pixel having value z is

$$p(z) = \frac{P_1}{\sqrt{2\pi}\sigma_1}\exp\left[-\frac{(z-u_1)^2}{2\sigma_1^2}\right] + \frac{P_2}{\sqrt{2\pi}\sigma_2}\exp\left[-\frac{(z-u_2)^2}{2\sigma_2^2}\right] \quad (1)$$

where μ_1 and μ_2 are the mean values of ROI and background, σ_1 and σ_2 are the standard deviations about the means, and P_1 and P_2 are the a priori probabilities of the two labels (ROI and background) and they are subject to $P_1 + P_2 = 1$. In this paper, P_1 and P_2 are the percentages of ROI and background pixels in an image (the areas of ROI and background divided by the area of the image). It is easy to show that if all the parameters (μ_1, μ_2, σ_1, σ_2, P_1 and P_2) are known, the optimal threshold that minimizes the misclassification of both ROI and background pixels is one of the roots of equation (Gonzalez and Woods, 1993)

$$AT^2 + BT + C = 0$$

where

$$A = \sigma_1^2 - \sigma_2^2$$
$$B = 2(\mu_1\sigma_2^2 - \mu_2\sigma_1^2) \quad (2)$$
$$C = \sigma_1^2\mu_2^2 - \sigma_2^2\mu_1^2 + \sigma_1^2\sigma_2^2 \ln\left(\frac{\sigma_2 P_1}{\sigma_1 P_2}\right)$$

In this paper, we use training images (except in river vs. field case) to obtain the optimal threshold and apply it to the testing images. Since we have the ground-truth for the training images, it is straightforward to determine P_1 and P_2, the means and standard deviations of the ROI and the background of training images. This algorithm requires that the distributions of pixel values within the ROI and the background of an image can be approximated by two Gaussian functions and it also implies that the ROIs account for the same or similar percentage of total image area in both training and testing images. However, sometimes these requirements may not be met (this implies that Eq. (1), commonly used for visual images, is not valid for SAR images and the assumption of Gaussian distribution for SAR images is not correct). For *example 1*—river vs. field, Eq. (2) corresponding to the training image has no real root, thus we use the testing image to determine the optimal threshold and apply it to the training image.

Fig. 11 shows the ROIs extracted by this algorithm. The fitness values of the extracted ROIs and their corresponding threshold values are also shown. From Fig. 11, it is obvious that the agents learned by normal GP and smart GP have better performance in ROI extraction. However, it takes the second traditional algorithm much less time to extract ROIs (<1 s on Ultra 2 Sun Workstation), since the threshold can be determined very quickly by determining the parameters needed to compute A, B and C and solving Eq. (2).

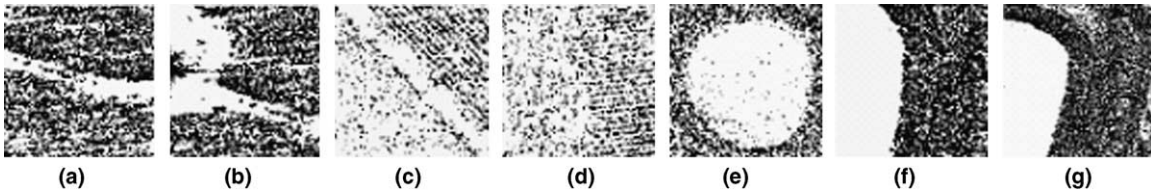

Fig. 11. ROI extracted (white pixels) by the traditional ROI extraction algorithm: (a) river vs. field (fitness = 0.24, threshold = 65.3); (b) river vs. field (fitness = 0.49, threshold = 65.3); (c) field vs. grass (fitness = 0.63, threshold = 99.5); (d) field vs. grass (fitness = 0.50, threshold = 99.5); (e) lake vs. field (fitness = 0.72, threshold = 37.8); (f) lake vs. grass (fitness = 0.73, threshold = 37.8); (g) lake vs. field (fitness = 0.63, threshold = 37.8).

5. Conclusions

In this paper, we presented a basic approach that uses genetic programming to evolve agents for extracting terrain regions in remotely sensed images. In order to improve the efficiency of genetic programming, we proposed smart crossover and smart mutation that can identify and keep the effective ROI extraction components in agents. We used SAR imagery to demonstrate the approach for extracting terrain regions. Our experimental results showed that GP can find good agents to effectively extract ROIs in SAR images and the smart crossover and smart mutation make GP find these good agents more quickly, thus greatly improving the efficiency of GP. The agents generated by GP during the training can be applied to extract ROIs in other similar images. Currently, in order to get the fitness at each node, we have to evaluate its output image, which is a time consuming and inefficient process. To further improve the efficiency of GP, it is important to find a way to estimate the fitness of internal nodes based on the fitness of the root node.

Acknowledgements

This research is supported by AFRL grant F33615-99-C-1440 and NSF grant IIS-0114036. The contents of the information do not necessarily reflect the position or policy of the US Government.

References

Acqua, F., Gamba, P., 2001. Detection of urban structures in SAR images by robust fuzzy clustering algorithms: The example of street tracking. IEEE Trans. Geosci. Remote Sensing 39 (10), 2287–2297.

Banzhaf, W., Nordin, P., Keller, R., Francone, F., 1998. Genetic Programming—An Introduction on the Automatic Evolution. Morgan Kaufmann Publishers.

Bhanu, B., Lin, Y., 2002. Learning composite operators for object detection. In: Proc. Genetic and Evolutionary Computation Conf., July, pp. 1003–1010.

D'haeseleer, P., 1994. Context preserving crossover in genetic programming. In: Proc. IEEE World Congress on Computational Intelligence, vol. 1, pp. 256–261.

Dong, Y., Milne, A., Forster, B., 2001. Segmentation and classification of vegetated areas using polarimetric SAR image data. IEEE Trans. Geosci. Remote Sensing 39 (2), 321–329.

Gonzalez, R., Woods, R., 1993. Digital Image Processing. Addison-Wesley Publishing Company.

Howard, D., Roberts, S.C., Brankin, R., 1999. Target detection in SAR imagery by genetic programming. Adv. Engrg. Software 30 (5), 303–311.

Ito, T., Iba, H., Sato, S., 1998. Depth-dependent crossover for genetic programming. In: Proc. IEEE Internat. Conf. on Evolutionary Computation, pp. 775–780.

Jeon, B., Jang, J., Hong, K., 2002. Road detection in spaceborne SAR images using a genetic algorithm. IEEE Trans. Geosci. Remote Sensing 40 (1), 22–29.

Katartzis, A., Sahli, H., Pizurica, V., Cornelis, J., 2001. A model-based approach to the automatic extraction of linear features from airborne images. IEEE Trans. Geosci. Remote Sensing 39 (9), 2073–2079.

Koza, J.R., 1994. Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press.

Mvogo, J., Onana, V., Rudant, J., Trebossen, H., Mercier, G., Tonye, E., 2000. Coastline detection in SAR images using wavelet packets, multiscale segmentation and a Markov random field regularization. In: Proc. SPIE-Conf. on SAR Image Analysis, Modeling, and Techniques III, vol. 4173, pp. 122–131.

Nordin, P., Banzhaf, W., 1995. Complexity compression and evolution. In: Proc. Sixth Internat. Conf. on Genetic Algorithms, pp. 310–317.

Poli, R., 1996. Genetic image segmentation. In: T.C. Forgarty (Ed.), Evolutionary Computation, pp. 110–125.

Roberts, S.C., Howard, D., 1999. Evolution of vehicle detectors for infrared line scan imagery. In: Proc. Wkshp., Evolutionary Image Analysis, Signal Proc. and Telecomm., pp. 110–125.

Segl, K., Kaufmann, H., 2001. Detection of small objects from high-resolution panchromatic satellite imagery based on supervised image segmentation. IEEE Trans. Geosci. Remote Sensing 39 (9), 2080–2083.

Smith, P., 1996. Conjugation—A bacterially inspired form of genetic recombination. In: Koza, J.R. (Ed.), Late Breaking Papers at the Genetic Programming Conf., pp. 167–176.

Stanhope, S.A., Daida, J.M., 1998. Genetic programming for automatic target classification and recognition in synthetic aperture radar imagery. In: Proc. Conf. Evolutionary Programming VII, pp. 735–744.

Tackett, W., 1994. Recombination. selection, and the genetic construction of computer programs, Ph.D. thesis, Department of Electr. Engg. Systems, University of Southern California.