Visual Learning by Evolutionary Feature Synthesis

Krzysztof Krawiec[1] KKRAWIEC@VISLAB.UCR.EDU
Bir Bhanu BHANU@CRIS.UCR.EDU
Center for Research in Intelligent Systems, University of California, Riverside, CA 92521, USA

Abstract

In this paper, we present a novel method for learning complex concepts/hypotheses directly from raw training data. The task addressed here concerns data-driven synthesis of recognition procedures for real-world object recognition task. The method uses linear genetic programming to encode potential solutions expressed in terms of elementary operations, and handles the complexity of the learning task by applying cooperative coevolution to decompose the problem automatically. The training consists in coevolving feature extraction procedures, each being a sequence of elementary image processing and feature extraction operations. Extensive experimental results show that the approach attains competitive performance for 3-D object recognition in real synthetic aperture radar (SAR) imagery.

1. Introduction

Visual learning is a challenging domain for machine learning (ML) for several reasons. Firstly, visual learning is a *complex* task, that usually requires problem decomposition, which is nontrivial in itself. Secondly, the visual training data is *represented* in a way that is inconvenient for most standard ML methods, and requires use of specialized procedures and operators to access, aggregate, and transform the input. Thirdly, the *amount of data* that have to be processed during the training process is usually much higher than in standard ML applications. This imposes significant constraints on the effectiveness of the hypothesis space search. Finally, the real-world image data is usually *noisy* and contains plenty of *irrelevant* components that have to be sieved out in the learning process.

The approach for recognizing objects in real-world images described in this paper addresses *all* these issues and attempts to solve these problems by using important ideas from machine learning, evolutionary computation (EC), and computer vision (CV), and combining them in a novel way.

2. Motivation, Related Work and Contribution

2.1 Motivation

The primary motivation for the research described in this paper is the lack of general methodology for the design and development recognition systems. The design of recognition system for most real-world tasks is tedious, time-consuming and expensive. Though satisfactory in performance in constrained situations, the handcrafted solutions are usually limited in scope of applicability and have poor adaptation ability in practical applications. As the complexity of the task of object recognition by computer increases, the above limitations become severe obstacles for the development of solutions to real-world problems. In some aspects, it is similar to the way the complexity of the software development process made the developers struggle until the software engineering came into being.

2.2 Related Work

The interest in visual learning research has been rather limited in both ML and CV communities, although the importance of vision in the development of intelligent systems has been well recognized. In most approaches reported in the literature, adaptation is limited to parameter optimization that usually concerns a particular processing step, such as image segmentation, feature extraction, etc. In those cases, learning does affect the overall recognition result in some complex manner.

Current recognition systems are mostly open-loop and human input in the design of these systems is still predominant. Only a few contributions, summarized in Table 1, attempt to close the feedback loop of the learning process at the highest (e.g., recognition) level *and* test the proposed approach in real-world setting. Note that, to the best of our knowledge, only few approaches (Teller & Veloso, 1997; Peng & Bhanu, 1998a; Peng & Bhanu,

[1] On a temporary leave from Poznan University of Technology, Poznan, Poland.

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

Table 1. Related work in visual learning.

Reference	Approach	Experimental task	Training data
(Draper, 1993)	Learning recognition graphs	Recognizing buildings	Higher-level CV concepts
(Segen, 1994)	Learning of object models	Hand gesture recognition	Graphs extracted from images
(Johnson, 1995)	EC (GP)	Locating hand in human body silhouette	Binary silhouettes
(Teller & Veloso, 1997)	EC (GP variant)	Face recognition	Raw images (grayscale)
(Peng & Bhanu, 1998a)	Reinforcement learning	Segmentation of in/outdoor scenes	Raw images (color)
(Peng & Bhanu, 1998b)	Delayed reinforcement learning	Segmentation and feature extraction, in/outdoor	Raw images (color)
(Krawiec, 2001)	EC (GP)	Handwritting recognition	Raw images (grayscale)
(Rizky et. al., 2002)	Hybrid EC (GP+NN)	Target recognition in radar modality	1-D radar signals
(Maloof et. al., 2003)	Standard ML/PR classifiers	Rooftop detection in aerial imagery	Fixed set of scalar features
This contribution	EC (CC+LGP)	Object recognition in radar modality	Raw image (grayscale)

1998b; Krawiec, 2001) have been reported that learn using *raw* images as training data, and, therefore, produce the *entire* object recognition system. Moreover, a majority of these methods (Segen, 1994; Johnson, 1995; Maloof, 2003) use domain-specific knowledge and are highly specialized towards a particular application.

2.3 Contributions of this Paper

(a) We propose a *general approach to automatic learning/synthesis of recognition procedures*, that (i) uses raw image data for training, (ii) does not require domain-specific knowledge, and (iii) attains competitive performance on a complex, real-world object recognition task. The learning proceeds given only database of training examples (images) partitioned into decision classes, and a set of general-purpose image processing and feature extraction operators. We use the cooperative coevolution (Potter & De Jong, 2000), a new paradigm of EC, to handle the complexity of the task.

(b) We use EC to perform the visual learning meant as the search in the space of image representations (features).

(c) We adopt a variety of linear genetic programming (LGP) (Banzhaff et. al., 1998) for encoding of basic image processing and feature extraction procedures.

(d) We use the real image data to demonstrate our approach and provide a comparison of performance between the coevolutionary approach and standard GA.

3. Technical Approach

The proposed approach operates in a learning-from-examples scheme, with learner/inducer autonomously acquiring knowledge from the training examples (images). The output of the learner is the synthesized recognition system, that implements the *feature-based recognition* paradigm, with processing split into two stages: *feature extraction* and *decision making*. In particular, we include the image processing and feature extraction into the learning process (learning loop). The learner is, therefore, able to design the intermediate image representation that is appropriate for solving the task faced. Note that, from machine learning viewpoint, this approach may be regarded as a kind of constructive induction (Matheus, 1989).

3.1 Evolving Recognition Procedures

The learning proceeds in the framework of evolutionary computation, where we evolve *procedures* being sequences of elementary image processing and feature extraction operations. The evolutionary algorithm maintains a set of such procedures that are modified and mated during the evolutionary search (Fig. 1). The procedures compete with each other by means of their fitness values that reflect the utility of particular representation for solving the problem. The best procedure found in the evolutionary run becomes the final result of the procedure synthesis.

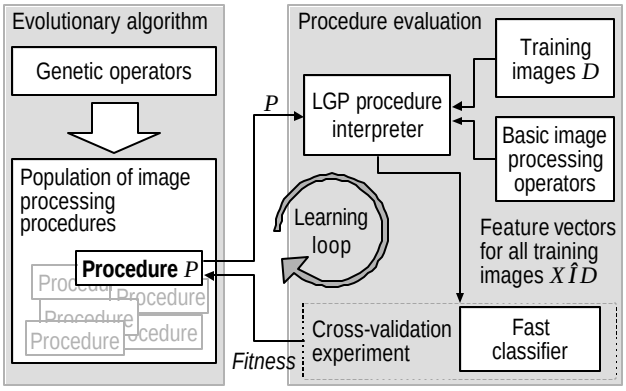

Figure 1. The overall architecture of our learning system.

3.2 Representation of Feature Extraction Procedures

An important issue that influences the performance of the proposed approach is the representation of individuals. To speed up the convergence of the search process and provide the system with basic knowledge, we assume that certain elementary building blocks are given *a priori* to

the learner in a form of basic image processing, feature extraction, and feature transformation operators.

A variety of linear genetic programming (LGP) (Banzhaff et. al., 1998) is chosen as the representation framework for the described system. LGP is a hybrid of genetic algorithms (GA), and genetic programming (GP). The LGP *genome*, i.e. the internal encoding of solution, is a fixed-length string of numbers that is interpreted as a sequential procedure. The procedure is composed of (possibly parameterized) basic operations that work on input data/images. The major advantage of this linear representation is low susceptibility to destructive crossovers, which is an important problem in GP.

The details of LGP procedure encoding may be briefly summarized as follows:

- Each procedure *P* is a fixed-length string of bytes [0..255] that encodes sequence of *operators*, i.e. image processing and feature extraction algorithms.
- The operations work on *registers* (working variables) used for both input and output during procedure execution. *Image registers* store processed images, whereas *real-number registers* store scalar features. All image registers have the same dimensions as the input image. Each image register, apart from storing the image, maintains a single rectangular mask. A single learning parameter n_{reg} controls both the number of image and number registers.
- Each chunk of 4 consecutive bytes in the LGP procedure encodes a single operation with the following elements: (i) operation code, (ii) mask flag – decides whether the operation should be global (work on the entire image) or local (limited to the mask), (iii) mask dimensions (ignored if mask flag is 'off'), (iv) arguments – numbers (identifiers) of registers to fetch input data and store the result.

An example of operation is morphological opening (operation code) using rectangular ROI (ROI flag 'on') of size 14 (ROI size) on the image fetched from image register #4 (pointed by argument #1), and storing the result in image register #5 (pointed by argument #2).

There are currently approx. 70 operations implemented in the system, consisting mostly of Intel Image Processing (Intel Corp., 2000) and OpenCV (Intel Corp., 2001) libraries. They may be grouped into following categories: image processing operations, mask – related operations, feature extraction operations, and arithmetic and logic operations.

Given the above settings, an LGP procedure *P* processes a single input image *I* in the following steps (see Fig. 2):

1. *Initialization* of register contents: Each of the n_{reg} image registers is set to *I*. The masks of images are set to consecutive local features (here: bright 'blobs') found in the image, so that mask in the i^{th} image register encompasses i^{th} local feature. Real-number registers are set to the midpoint coordinates of corresponding masks; in particular, real-number registers $2i$ and $2i+1$ store the x and y coordinates of the i^{th} image mask.
2. *Execution*: the operations encoded by *P* are carried out one by one. As a result, the contents of image and real-number registers change (see example in Fig. 2).
3. *Interpretation*: the values computed and stored in the real-value registers are interpreted as the output yielded by *P* for image *I*. Let us denote by $f_i(P,I)$ the value stored by *P* in the real-value register #*i* when processing image *I*. Then, for an image *I*, the LGP procedure outputs a vector of features:

$$\left\langle f_1(P,I), f_2(P,I), \ldots, f_{n_{reg}}(P,I) \right\rangle$$

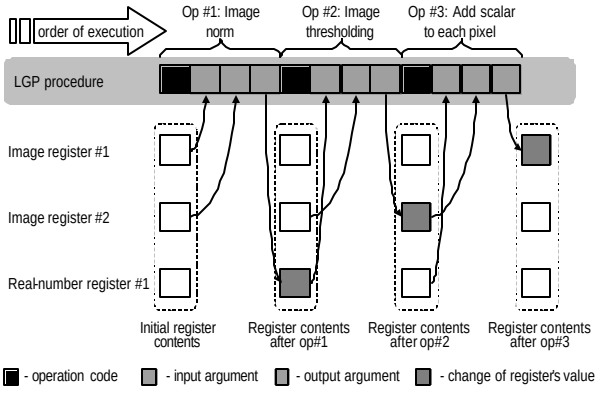

Figure 2 Illustration of the process of genome interpretation during LGP procedure execution.

3.3 Cooperative Coevolution

To cope with the *inherent complexity* of the visual learning task, we should find a way to *decompose the problem* into subtasks rather than trying to solve it in one step. For that purpose, we use the cooperative coevolution, a variety of evolutionary computation.

Evolutionary computation is widely recognized as a kind of *metaheuristics*, i.e. general-purpose search algorithm that provides suboptimal solutions in polynomial time. However, according to Wolpert's 'no free lunch' theorem (Wolpert & Macready, 1997), the search for an universal, best-of-all metaheuristic (optimization or learning) algorithm is futile. In other words, the average performance of any metaheuristic over a set of all possible fitness functions is the same.

In real world however, not all fitness functions are equally probable. Most real problems are characterized by some *features* that make them specific. The practical utility of a search/learning algorithm depends, therefore, on its ability to detect and benefit from that specificity. In particular, the *complexity* of the problem and the way it may be *decomposed* are such characteristics.

In the last few years, *cooperative coevolution* (CC) (Potter & De Jong, 2000), a variety of EC, has been reported as a promising approach to handle the increasing complexity of problems posed in artificial intelligence and related disciplines. There are two important factors that make CC different from standard EC. Firstly, instead of having just one population of individuals, in CC one maintains *many* of them. Secondly, individuals in particular population encode only *part* of the solution to the problem, as opposed to EC, where each individual encodes complete solution to the problem. Therefore, individuals from populations cannot be evaluated independently; they have to be combined with some *representatives* from the remaining populations to form a solution that can be evaluated. That is why evolution proceeds here in each population independently, with the exception of the evaluation stage. The joint evaluation scheme forces the individuals from particular populations to cooperate.

Let n denote the number of populations. To evaluate an individual X from i^{th} population (Fig. 3), it is temporarily combined with selected individuals (so called *representatives*) from the remaining populations j, j=1,…, n, $j \neq i$, to form the solution. Then, the entire solution is evaluated by means of the fitness function and X gets the resulting fitness value. Evaluation of an individual from i^{th} population does not affect the remaining populations.

```
initialize populations
loop
    for each population
        for each individual X
            combine X with representatives of
                remaining populations to form solution S
            evaluate S and assign its fitness to X
        end for
        select mating candidates
        mate parents; use their offspring as next generation
    end for
until stopping condition
return best solution
```

Figure 3. Outline of cooperative coevolution algorithm.

As a result, the evolutionary search in a given population is driven by the context build up by the representatives of remaining populations. The choice of representatives is, therefore, critical for the convergence of the evolution process. Although many different variants are possible here, it has been shown that so-called CCA-1 scheme works best (Wiegand, Liles, & De Jong, 2001). In the first generation a representative of i^{th} population is an individual drawn randomly from it. In the following generations a representative of i^{th} population is the best individual w.r.t. the previous generation.

The major advantage of CC is that it provides the possibility of breaking up a complex problem into components *without specifying explicitly the objectives for them.* The way the individuals from populations cooperate *emerges* as the evolution proceeds. In (Bhanu & Krawiec, 2002) we provided experimental evidence for the usefulness of CC in feature construction for standard machine learning problems. Here we claim that CC is especially appealing also to the problem of visual learning, where the overall target is well defined, but there is no *a priori* knowledge about what should be expected at intermediate stages of processing, or such knowledge requires an extra effort from the designer.

3.4 Combining Cooperative Coevolution and Linear Genetic Programming

In the proposed approach, we use cooperative coevolution to scale down the task of LGP procedure synthesis (Section 3.2). Although this can be done in many different ways, in this initial contribution we break up the task at *genome level*, with each population being responsible for optimizing a pre-defined fragment (substring) of LGP code of fixed length (Fig. 4).

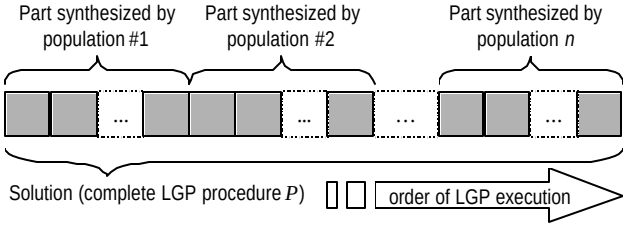

Figure 4. Cooperation enforced by the concatenation of LGP procedure fragments developed by particular populations.

The evaluation of an individual X from a given population consists in concatenating (always in the same order) its genome with the genomes of the representatives of the remaining populations to form a single LGP procedure P. P is then executed for all images from the training set (see Section 3.2). The values computed by P for all training images

$$\left\langle f_1(P,I),\ f_2(P,I),\ldots,\ f_{n_{reg}}(P,I)\right\rangle, \forall I \in T,$$

together with the images' class labels constitute the dataset T that is the basis for evaluation of an individual (so-called *fitness set*). Then, a fast classifier is trained and tested on these data (see Fig. 1), using predefined internal division of the training set into training-training set and training-testing set. For this purpose, we used the naïve Bayesian classifier, modeling the input variables (features) by normal distribution. The resulting predictive *recognition ratio*,

$$\frac{\#\text{ of correctly classified objects from } T}{\text{total}\#\text{ of images in } T},$$

becomes the evaluation (fitness) of the solution-procedure P, and is subsequently assigned to the individual X.

In this framework, particular populations can specialize in different stages of the recognition task. In particular, we expect that the populations delegated to the development of the early parts of LGP procedure would tend to specialize in image processing, whereas the populations working on the final parts of the LGP procedure would focus on feature extraction and aggregation.

4. Experiments

The objective of the computational experiments is to explore the overall idea of LGP-based synthesis of recognition procedures using cooperative coevolution for search, in the context of demanding, real-world object recognition task using images of 3-D objects. The results are obtained using a PC with single Pentium 1.8 GHz processor.

To provide a reference solution, we run a separate series of standard linear genetic programming (LGP), which, in fact, is a special case of CC that uses just one population. To make this comparison reliable, we *fix the total genome length* (the total procedure length is the same for both CC and standard LGP), and *fix the total number of individuals* (the total number of individuals from all populations in CC is equal to the number of individuals maintained in the single population of the corresponding LGP run). To estimate the performance the learning algorithm is able to attain in a limited time, evolution stops when its run time reaches the predefined limit.

4.1 Parameter Setting

The following parameter setting has been used in the experiments: mutation: one-point, prob. 0.5; crossover: one-point, prob. 1.0, genome cutting is allowed at every point; selection operator: tournament selection with pool size = 5; number of registers (image and numeric) n_{reg}: 8; number of populations n: 3; selection of representatives: CCA-1 (see Section 3.3); time limit: 1000 and 2000 seconds; procedure length (total genome length): 72 bytes, i.e., 18 operations; total population size: 300 - 900 individuals. All the remaining parameters were set to default values used in software packages ECJ (Luke, 2002) and WEKA (Witten & Frank, 1999).

4.2 Data and the Learning Task

The proposed approach has been tested on the demanding task of object recognition in synthetic aperture radar (SAR) imagery. The MSTAR public database (Ross et. al., 1998) of SAR images taken at one foot resolution has been used as the data source. The task posed to the system was to recognize three different objects (decision classes): BRDM2, D7, and T62 (see Fig. 4) at 15° depression angle and any azimuth (0°-359°).

The difficulties associated with the object recognition task in real SAR images are:

- Non-literal nature of the data, i.e. radar images appear different than visual ones. Bright spots on the images, called scattering centers, correspond to those parts of the object which reflect radar signal strongly. No line features are present for these man-made objects at this resolution.
- Low persistence of features under rotation (high rotation-variance).
- High levels of noise.

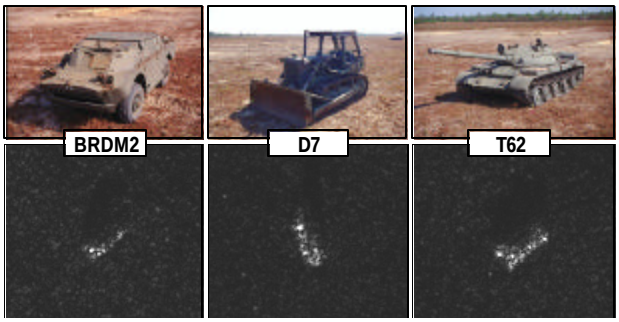

Figure 5. The representatives of three decision classes. Top row – visual photographs, bottom row - corresponding 48×48 pixel SAR images.

From the MSTAR database, 507 images of three objects classes (see Fig. 5) have been selected. The resulting set of images has been split into disjoint training and testing parts to provide reliable estimate of the recognition ratio of the learned recognition system (see Table 2). This selection was aimed at providing uniform coverage of the azimuth (for each class, there is a training image for approx. every 5.62° of azimuth, and a testing image every 2.9°-5.37°, on the average).

Table 2. Dataset statistics.

	Number of images				
Class	*Total*	*Training set*	*Aspect interval*	*Testing set*	*Aspect interval*
BRDM2	188	64	5.62°	124	2.90°
D7	188	64	5.62°	124	2.90°
T62	131	64	5.62°	67	5.37°
Total	507	192		315	

The evolutionary process uses the training data for the learning/synthesis (precisely speaking, for the fitness computation), whereas the testing images are used for test only. The original images have different sizes, so they are cropped to 48×48 pixels. They are also complex (2-channel), but only theirs magnitude part is used in the experiments. No other form of preprocessing (e.g., speckle removal) is applied.

4.3 Results

Table 3 compares the recognition performances obtained by the proposed coevolutionary approach (CC) and its

Table 3. The average performances of best individuals evolved in 10 independent runs for 1000 and 2000 seconds training time limit.

Method	*# populations*	*Parameter setting*				*Recognition ratio*			
		Procedure length		*# of individuals*		*1000 seconds*		*2000 seconds*	
		Each population	*Total*	*Each population*	*Total*	*Train set*	*Test set*	*Train set*	*Test set*
CC	3	24	72	100	300	0.915	0.867	0.933	0.890
LGP	1	72	72	300	300	0.806	0.747	0.843	0.801
CC	3	24	72	300	900	0.927	0.874	0.940	0.883
LGP	1	72	72	900	900	0.839	0.795	0.881	0.830

regular counterpart (LGP), for two different limits imposed on the evolutionary learning time, 1000 and 2000 seconds. To obtain statistical evidence, all evolutionary runs have been repeated 10 times, so the table presents the average performances of the best individuals found.

The direct comparison resulting from Table 3 shows the superiority of the CC to LGP. This applies to both the performance of the synthesized systems on the training as well as on the test set. In all cases, the observed increases in accuracy are statistically significant with respect to the one-sided t-Student test at the confidence level 0.05. Note that, within the same time limit, CC usually ran for a *smaller* number of generations on the average, due to the extra time required to maintain (perform selection and mating) in multiple populations.

Figure 6 and Table 4 show, respectively, the receiver operating characteristics (ROC) curves and confusion matrices for the best individuals found in the first two experiments reported in Table 3 (time limit: 2000 seconds, procedure length: 72, total # of individuals: 300). Each curve shows the true positive ratio, i.e., the share of correctly recognized objects, as a function of false positive ratio, i.e., the share of incorrectly classified objects (without taking into account the non-recognized objects).

These parametric characteristics have been obtained from the test set, by varying the confidence threshold of the naïve Bayesian classifier. Approximately 40 different values of the threshold have been used to obtain the curves. The confidence threshold imposes a lower limit on the ratio of *a posteriori* probabilities of the first and the second most probable decision classes. If, for a particular test example, the ratio is lower than threshold, no recognition decision is made and the example remains unclassified. The ROC curves clearly show the superiority of the coevolution. For instance, when no more than 5% of false positives is allowed, the procedure evolved using CC recognizes correctly approximately 91% images, whereas for LGP the accuracy is around 68%.

Figure 7 presents the processing carried out on a BDRM2 image (taken at 342.3° aspect) by the best procedure found in one of evolutionary runs. For clarity, the picture shows the interpretation of the LGP procedure in a form of data-flow graph. Each column of images shows the content changes of particular image register. Note that this procedure uses only first four of the total of eight image registers available, and, although the registers have initially the same contents (the input SAR image), their mask positions (small red squares) are different.

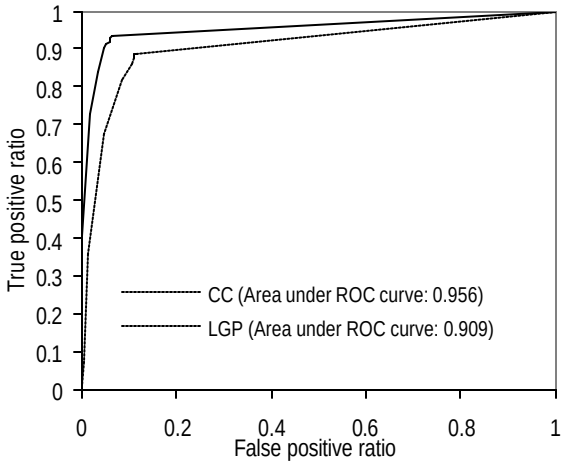

Figure 6. ROC curves obtained for the test set using the best individuals found in the first two experiments shown in Table 3.

The execution of the LGP procedure starts from the top and proceeds downwards through several intermediate image processing steps. Rounded and slanted boxes denote global (working on the entire image) and local (working on the marked rectangular ROI mask) image processing operations, respectively. Eventually, two of the executed operations yield scalar features (the x coordinate of the shifted ROI ($f_1(X,I)$), and the normalized difference of two processed images $f_2(X,I)$). The overall processing ends with the final recognition decision made by the (previously trained on the training set) classifier; this includes *a posteriori* probabilities yielded by the naïve Bayesian classifier.

The operations used in this particular are: *AbsDiff* – pixel-wise absolute difference of a pair of images, *HiPass3x3* – high pass convolution filter using 3×3 mask, *CrossCorrel* – cross-correlation of a pair of images, *PushROIX* – (local) shifts the current image's ROI to the closest bright 'blob' in horizontal direction, *Gaussian* – (local) image smoothing using 3×3 Gaussian mask, *MorphClose* – morphological closing operation, *LogicalOr* – pixel-wise logical 'OR' operation. Note that, commonly for genetic

programming, not all input data (initial register contents) and not all intermediate results are utilized for the final decision making (e.g., the result of the cross-correlation operation (*CrossCorrel*) is not further processed).

Table 4. Confusion matrices for the test set using the best individuals found in the first two experiments shown in Table 3.

CC	*Predicted class*			
Actual class	*BRDM2*	*D7*	*T62*	*None*
BRDM2	**118**	1	4	1
D7	5	**114**	3	2
T62	5	1	**61**	0

LGP	*Predicted class*			
Actual class	*BRDM2*	*D7*	*T62*	*None*
BRDM2	**97**	3	22	2
D7	0	**115**	9	0
T62	1	0	**66**	0

5. Conclusions

In this paper, we proposed a general evolutionary learning method that enables the learner to *acquire knowledge from complex/structural examples by autonomously transforming the input representation*. The described formulation of feature construction addresses two important issues. (1) The elementary operations give the learner an access to complex, structural input data that otherwise could not be directly used. (2) By incorporating the feature synthesis into the learning loop, the learner searches for performance improvement by modifying the input representation.

In experimental part, we provided an evidence for the possibility of solving, using the proposed approach, a demanding real-world task of *visual learning*. The encouraging results for SAR object recognition have been obtained without recurring to means that are commonly used in conventional approaches to the design of recognition systems, such as resorting to the database of object models, explicit estimation of object pose, hand-tuning of basic operations for a specific application, and, in particular, SAR-specific concepts or features like 'scattering center'. The obtained recognition ratios are also comparable to those achieved by standard methods.

Our approach learns in a *fully automatic manner*, and, therefore, at a little expense of human labor and expertise. The learning process requires only training data that is usually easy to acquire, i.e. images and their class labels, and does not rely on domain-specific knowledge, using only general vision-related knowledge encoded in basic operations. The objectivity of the learning process makes the results free from subjective flaws and biases, which the human-designed solutions are prone to.

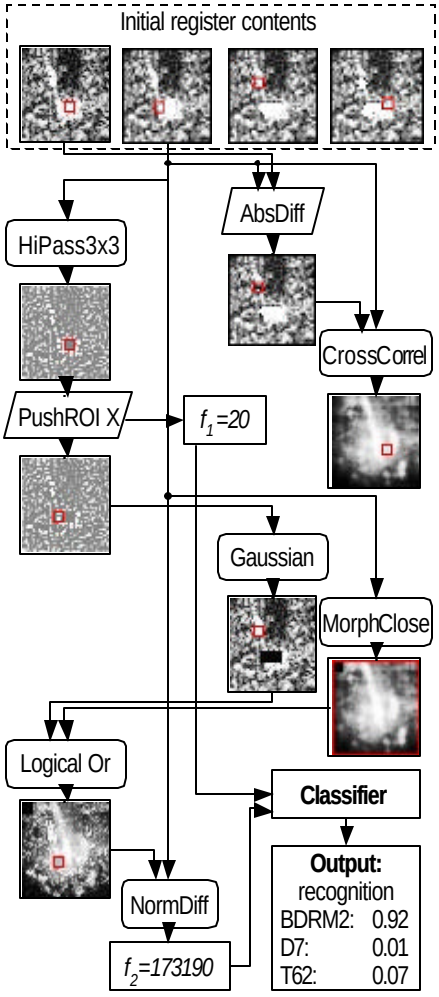

Figure 7. A fragment of synthesized processing graph of a selected best-of-run procedure evolved by means of cooperative coevolution, processing an exemplary image (only 4 of total 8 registers are used by this procedure).

The proposed method may be characterized as *feature-based*. Compared to the model-based recognition approaches, there is no need for, possibly expensive, matching an image with models from the database. Thus, our synthesized recognition system attains *high recognition speed* during the runtime. The average time required by the entire recognition process, starting from the raw image and ending up with the final recognition result, totaled 4.9 ms on the average, for a single 48×48 image and an LGP procedure composed of 18 operations. This time could be significantly reduced after re-implementing the synthesized system and, in particular, the classifier written in Java. We claim that this impressive recognition speed makes our approach suitable for real-time application.

Since the task-related knowledge is not required, our approach is general and possibly applicable to other recognition tasks and different image modalities. We

claim that, therefore, a new paradigm for visual learning has been developed, that focuses on automatic learning of pattern analysis procedures composed of relatively simple, general-purpose image processing and feature extraction building blocks, as opposed to the tendency of designing highly specialized procedures for particular recognition tasks.

From machine learning viewpoint, this result is an outstanding argument in favor of CC for tackling complex learning problems. The ability of coevolution to break up complex problems into subproblems without requiring explicit objectives/goals for them, offers an interesting research direction for ML.

Acknowledgements

We would like to thank the authors of software packages: ECJ (Luke, 2002) and WEKA (Witten & Frank, 1999) for making their software publicly available. This research was supported by the grant F33615-99-C-1440. The contents of the information do not necessarily reflect the position or policy of the U. S. Government. The first author is supported by the Polish State Committee for Scientific Research, research grant no. 8T11F 006 19.

References

Banzhaf, W., Nordin, P., Keller, R., & Francone, F. (1998). *Genetic programming. An introduction. On the automatic evolution of computer programs and its application*. San Francisco: Morgan Kaufmann.

Bhanu, B. & Krawiec, K. (2002). Coevolutionary construction of features for transformation of representation in machine learning. *Proceedings of Genetic and Evolutionary Computation Conference (Workshop on Coevolution)*. New York: AAAI Press, 249-254.

Draper, B., Hanson, A., & Riseman, E. (1993). Learning blackboard-based scheduling algorithms for computer vision. *International Journal of Pattern Recognition and Artificial Intelligence, 7*, 309-328.

Intel® image processing library: *Reference manual*. Intel Corporation, 2000.

Johnson, M.P. (1995). *Evolving visual routines*. Master's Thesis, Massachusetts Institute of Technology.

Krawiec, K. (2001). Pairwise comparison of hypotheses in evolutionary learning. In: C.E. Brodley, & A. Pohoreckyj Danyluk (Eds.), *Proceedings of the Eighteenth International Conference on Machine Learning*. San Francisco: Morgan Kaufmann, 266-273.

Luke, S. (2002). ECJ Evolutionary Computation System. http://www.cs.umd.edu/projects/plus/ec/ecj/.

Maloof, M.A., Langley, P., Binford, T.O., Nevatia, R., & Sage, S. (2003). Improved rooftop detection in aerial images with machine learning. *Machine Learning* (in press).

Matheus, C.J. (1989). A constructive induction framework. *Proceedings of the Sixth International Workshop on Machine Learning*. New York: Ithaca, 474-475.

Open source computer vision library: *Reference manual*. Intel Corporation, 2001.

Peng, J. & Bhanu, B. (1998a). Closed-loop object recognition using reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*, 139-154.

Peng, J. & Bhanu, B. (1998b). Delayed Reinforcement Learning for Adaptive Image Segmentation and Feature Extraction. *IEEE Transactions on Systems, Man and Cybernetics, 28*, 482-488, August 1998.

Poli, R. (1996). *Genetic programming for image analysis*. (Technical Report CSRP-96-1). University of Birmingham.

Potter, M.A. & De Jong, K.A. (2000). Cooperative coevolution: an architecture for evolving coadapted subcomponents. *Evolutionary Computation, 8*, 1-29.

Rizki, M., Zmuda, M., & Tamburino, L. (2002). Evolving pattern recognition systems. *IEEE Transactions on Evolutionary Computation, 6*, 594-609.

Ross, T., Worell, S., Velten, V., Mossing, J., & Bryant, M. (1998). Standard SAR ATR evaluation experiments using the MSTAR public release data set. *SPIE Proceedings: Algorithms for Synthetic Aperture Radar Imagery V*, Vol. 3370, Orlando, FL, 566-573.

Segen, J. (1994). GEST: A learning computer vision system that recognizes hand gestures. In R.S. Michalski, & G. Tecuci (Eds.), *Machine learning. A Multistrategy Approach*. Volume IV. San Francisco: Morgan Kaufmann, 621-634.

Teller, A. & Veloso, M. (1995). A controlled experiment: evolution for learning difficult image classification. *Lecture Notes in Computer Science, Vol. 990*, 165-185.

Wiegand, R.P., Liles, W.C., & De Jong, K.A. (2001). An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. *Proceedings of Genetic and Evolutionary Computation Conference*, San Francisco: Morgan Kaufmann, 1235-1242.

Witten, I.H. & Frank, E. (1999). *Data mining: Practical machine learning tools and techniques with Java implementations*, San Francisco: Morgan Kaufmann.

Wolpert, D. & Macready, W.G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*, 67-82.