Ensemble Image Classification Method Based on Genetic Image Network

Shiro Nakayama, Shinichi Shirakawa, Noriko Yata and Tomoharu Nagao

Graduate School of Environment and Information Sciences, Yokohama National University, 79-7, Tokiwadai, Hodogaya-ku, Yokohama, Kanagawa, 240-8501, Japan
shiro@nlab.sogo1.ynu.ac.jp, shirakawa@nlab.sogo1.ynu.ac.jp, yata@nlab.sogo1.ynu.ac.jp, nagao@ynu.ac.jp

Abstract. Automatic construction method for image classification algorithms have been required. Genetic Image Network for Image Classification (GIN-IC) is one of the methods that construct image classification algorithms automatically, and its effectiveness has already been proven. In our study, we try to improve the performance of GIN-IC with AdaBoost algorithm using GIN-IC as weak classifiers to complement with each other. We apply our proposed method to three types of image classification problems, and show the results in this paper. In our method, discrimination rates for training images and test images improved in the experiments compared with the previous method GIN-IC.

1 Introduction

Automatic construction method for image classification algorithms have been required. In general, image classification algorithms consist of image preprocessing, feature extraction, and classification process. It is very difficult to construct an algorithm suitable for all image classification problems. Therefore, a method is required to construct an image classification algorithm that would automatically adjust to the target problem is needed. Genetic Image Network for Image Classification (GIN-IC) [1] is one of the methods that construct image classification algorithms automatically, and its effectiveness has already been proven. GIN-IC automatically constructs the adequate classification algorithm (including image transformation, feature extraction and arithmetic operation components) using evolutionary computation. The process of GIN-IC is, first, to transform original images to easier-to-classify images using image transformation nodes, and next, to select adequate image features using feature extraction nodes. The greatest advantage of GIN-IC is its image transformation (preprocessing) component, which influences image feature selection. However, learning failure or over fitting of the training images sometimes occurs in the constructed algorithms because of GIN-IC's simple output to decide the classification.

In this paper, we extend GIN-IC by adding the AdaBoost algorithm [2]. AdaBoost is one of the greatest general ensemble learning methods to make a strong classifier, which has higher performance, by combining weak classifiers,

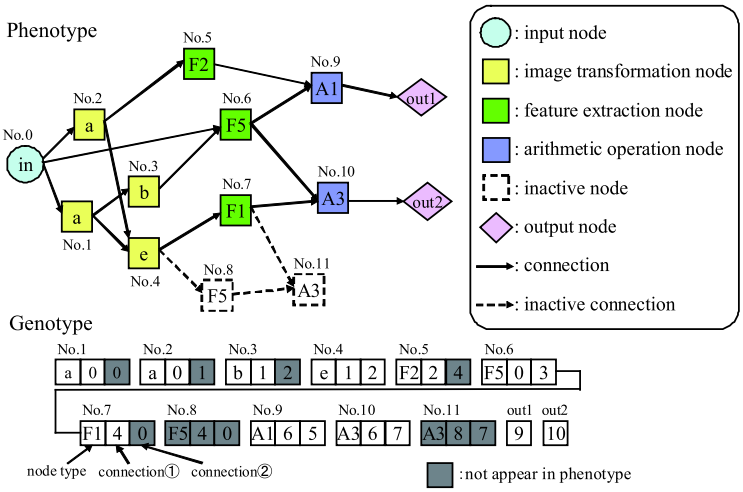

Fig. 1. Example of a structure in Genetic Image Network for Image Classification.

which have lower performance. AdaBoost is applied to various image classification problems and has shown its effectiveness, such as the face detection method proposed by Viola and Jones [3]. Moreover, aggregating the classifiers constructed by genetic programming [4–6], particle swarm optimization [7] or neural networks [8] has been studied to improve their classification performance. In our method, a set of output nodes in GIN-IC is treated as a weak classifier. The performance is expected to be improved by using GIN-IC as weak classifiers that complement with each other.

The next section of this paper is an overview of GIN-IC. In section 3, we describe our proposed method. In section 4, we apply the proposed method to three kinds of image classification problems and show their results in section 5. Finally, in section 6, we describe conclusions and future work.

2 Genetic Image Network for Image Classification (GIN-IC)

2.1 Structure of GIN-IC

The image classifier GIN-IC consists of image transformation, feature extraction, and arithmetic operation components based on the Genetic Image Network [9]. GIN-IC constructs an acyclic network-structured image classifier automatically. Fig. 1 shows an example of the phenotype (feed-forward network structure) and genotype (string representing the phenotype) of GIN-IC.

One of the benefits of this type of representation is that it allows the implicit reuse of nodes in its network. The nodes of GIN-IC are categorized into five types: input nodes, image transformation nodes, feature extraction nodes,

arithmetic operation nodes, and output nodes. Input nodes correspond to the original images. Image transformation nodes execute image transformation using the corresponding well-known image processing filters. Feature extraction nodes extract an image feature from the input images. Arithmetic operation nodes execute arithmetic operations. Image classification is performed using the values of the output nodes. In GIN-IC, these processes evolve simultaneously.

In GIN-IC, the feed-forward network structure of nodes is evolved, as shown in Fig. 1. Numbers are allocated to each node, beforehand. Increasingly large numbers are allocated, in order, to the input nodes, image transformation nodes, feature extraction nodes, arithmetic operation nodes, and output nodes. Connections, such as the feedback structure, that cannot be executed are restricted at the genotype level. The nodes take their input from the output of the previous nodes in a feed-forward manner. Because GIN-IC constructs a feed-forward network structured image classification procedure, it can represent multiple outputs. Therefore, GIN-IC enables easy construction of a multiclass image classification procedure using a single network structure.

To adopt an evolutionary method, GIN-IC uses genotype-phenotype mapping. This genotype-phenotype mapping method is similar to Cartesian Genetic Programming (CGP) [10]. The feed-forward network structure is encoded in the form of a linear string. The genotype in GIN-IC is a fixed length representation and consists of a string that encodes the node function ID and connections of each node in the network. However, the number of nodes in the phenotype can vary in a restricted manner, as not all the nodes encoded in the genotype have to be connected. This allows the existence of inactive nodes. In Fig. 1, node No. 8 and 11 are inactive nodes.

2.2 Genetic Operator and Generation Alternation Model

To obtain the optimum structure, an evolutionary method is adopted. The genotype of GIN-IC is a linear string. Therefore, it is able to use a standard genetic operator. In GIN-IC, mutation is used as the genetic operator. The mutation operator affects one individual, as follows:

- Select several genes randomly according to the mutation rate P_{m} for each gene.
- Randomly change the selected genes under the structural constraints.

(1 + 4) Evolution Strategy ((1 + 4) ES) is used as the generation alternation model. The (1 + 4) ES procedure in the experiments works as follows:

1. Set generation counter $j = 0$. Generate an individual randomly as a parent M.
2. Generate a set of four offspring C, by applying the mutation operation to M.
3. Select the elite individual from the set $M + C$ (the offspring is selected if it has the same best fitness as the parent). Then replace M with the elite individuals.

4. Stop if a certain specified condition is satisfied; otherwise, set $j = j + 1$ and go to step 2.

Since GIN-IC has inactive nodes, a neutral effect on fitness is caused by genetic operation (called neutrality [10]). In step 3, the offspring is selected if it has the same best fitness as the parent, then the searching point moves even if the fitness is not improved. Therefore, efficient search is achieved though a simple generation alternation model. This (1 + 4) ES was adopted and showed its effectiveness in previous works [1, 10].

2.3 Advantages and Limitations of GIN-IC

GIN-IC has many advantages to automatically construct image classification algorithms. The greatest advantage of GIN-IC is its image transformation (preprocessing) component, which is expected to influence image feature selection. In other words, GIN-IC generates and selects adequate image features by a combination of nodes. However, learning failure or over fitting of the training images sometimes occurs in the constructed algorithms because of GIN-IC's simple output to decide the classification. Moreover, more the size or the number of training images is increased, more time for learning it takes.

3 Proposed Method

3.1 Overview

To solve the problems noted in previous section, we apply the AdaBoost [2] algorithm to GIN-IC. A set of output nodes in GIN-IC is treated as a weak classifier. Thus, the total number of output nodes is $N \times T$, where N is the number of classes and T is the number of weak classifiers. Weak classifiers are evolved in sequence until each weak classifier achieves a specified error rate. In this process, the construction of the previous weak classifiers is fixed and can be reused in the subsequent weak classifiers. The effective process of other weak classifiers can be reused. Moreover, reuse is expected to reduce the time required for learning by avoiding recalculation at each operational node. In addition, all weak classifiers have a weight of their hypothesis. The final hypothesis is a weighted vote of the hypothesis of all weak classifiers. Fig. 2 shows an example of a structure in our proposed method for binary classification.

3.2 Process of the Proposed Method

The process of our proposed method is described as follows:

Step 1: Initialize the weights $D_1(i) = \frac{1}{m}$, $(i = 1, 2, \ldots, m)$, and weak classifier counter $t = 1$, where m is the number of training images.

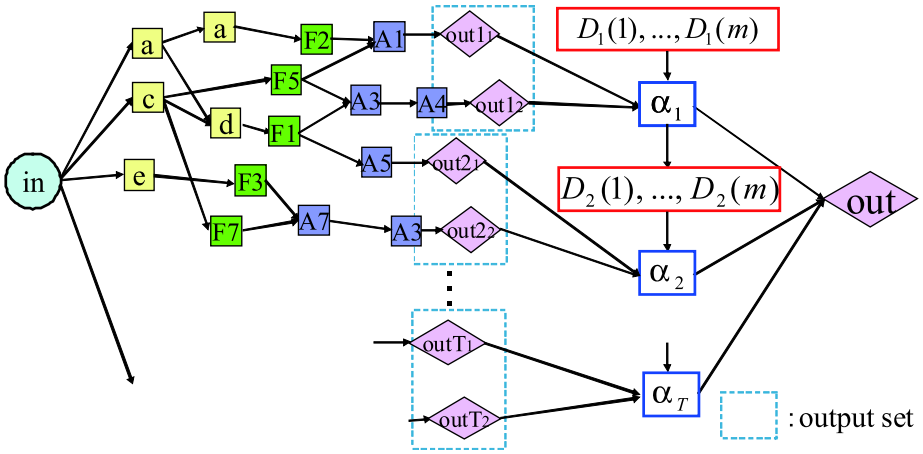

Fig. 2. Example of a structure in proposed method for binary classification.

Step 2: Focus the tth output set as the tth weak classifier. Here, add the new usable nodes to save enough nodes for constructing the new weak classifier. Add n_{i} of image transformation nodes, n_{f} of feature extraction nodes and n_{a} of arithmetic operation nodes.

Step 3: Evolve the weak classifier based on GIN-IC till the error rate ϵ_t is less than a threshold τ. ϵ_t is obtained by summing weights $D_t(i)$ corresponding to misclassified images as

$$\epsilon_t = \sum_{i:h_t(x_i)\neq y_i} D_t(i), \tag{1}$$

where $h_t(x_i)$ is the class label corresponding to the output node with the largest value of the weak classifier t when the input is image x_i, and y_i is the correct class label of image x_i. Here, we use $(1 + 4)$ ES as the generation alternation model. The $(1 + 4)$ ES procedure is the same as in section 2.2. Do not operate the fixed nodes in step 6 by genetic operation, and allow these nodes to be selected as the input of other nodes. When constructing the tth classifier, the structures of the classifiers from first to $(t-1)$th can be reused.

Step 4: Calculate the weight of hypothesis α of the focused output set as

$$\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right). \tag{2}$$

Step 5: Update the weights as

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \exp(-\alpha_t) & \text{if } h_t(x_i) = y_i \\ \exp(\alpha_t) & \text{otherwise} \end{cases}, \tag{3}$$

where Z_t is the number for normalization. This operation adapts the weights corresponding to the training images based on the AdaBoost algorithm. By this operation, the weights corresponding to the training images classified correctly are decreased and the weights corresponding the training images misclassified by the previous weak classifiers are increased. Therefore, the misclassified images are classified correctly by the next weak classifier preferentially.

Step 6: Set $t = t+1$. If $t \leq T$, fix the nodes connected with the focused output set and go back to step 2.

Step 7: Output the final hypothesis $h_{\text{fin}}(x_i)$ as

$$h_{\text{fin}}(x_i) = \arg\max_{y_i \in Y} \sum_{t:h_t(x_i)=y_i} \alpha_t. \tag{4}$$

This operation calculate the final classification results by a vote of all weak classifiers. For all weak classifiers, sum the weights of hypothesis α_t for the class output by tth weak classifier. The input image x_i is classified the class $h_{\text{fin}}(x_i)$, which maximizes the sum of the weights of weak classifiers that output the class.

3.3 Characteristics of the Proposed Method

We note some characteristics of our proposed method in this section. First, the classifier constructed by our proposed method consists of a number of GIN-IC as weak classifiers. We think that the performance is improved because weak classifiers complement with each other compared with GIN-IC. Second, the construction of the previous weak classifiers is fixed and can be reused in the subsequent weak classifiers. Therefore, the effective process of other weak classifiers can be reused since all nodes may be selected as the input of other nodes. Third, we stop evolving GIN-IC on the way to use it as weak classifiers. This operation is expected to reduce the time required for learning and to prevent over fitting.

4 Experiments

4.1 Settings of the Experiment

In this section, we evaluate our proposed method by applying it to three image classification problems. The problems are as follows:

1. Texture images
2. Pedestrian images
3. Generic object images

In these experiments, we transform all color images into grayscale. We also apply the previous method, GIN-IC, to the same problems to compare it with our proposed method. Table 1 shows the parameters used in the proposed method

Table 1. Parameters used in the experiments.

Parameters	GIN-IC	Proposal
Generation alternation model	(1+4)ES	(1+4)ES
Mutation rate (P_{m})	0.02	0.02
Image transformation nodes (n_{i})	100	Add 50
Feature extraction nodes (n_{f})	100	Add 50
Arithmetic operation nodes (n_{a})	100	Add 50
Output nodes	6, 2, 5	6, 2, 5
The number of generations	112500	–
The number of weak classifiers (T)	–	100
Error rate threshold (τ)	–	0.4

and the previous method, GIN-IC. We determined each parameter by the results of preliminary experiments.

We prepare simple and well-known image processing filters as the image transformation nodes in the experiments (26 one-input, one-output filters and 9 two-input, one-output filters), e.g., mean filter, maximum filter, minimum filter, Sobel filter, Laplacian filter, gamma correction filter, binarization, linear transformation, difference, logical sum, logical prod, etc. 17 simple statistical values are used as feature extraction nodes, e.g., mean value, standard deviation, maximum value, minimum value, mode, 3 sigma in rate, 3 sigma out rate, skewness, kurtosis, etc. The arithmetic operation nodes are 20 well-known arithmetic operations, e.g., addition, subtraction, multiplication, division, threshold function, piecewise linear function, sigmoid function, absolute value, equalities, inequalities, constant value, etc. In both the proposed method and previous method (GIN-IC), we use the same kinds of nodes. The fitness function of GIN-IC as weak classifiers in proposed method is described as follows:

$$\mathrm{fitness} = \mathrm{N_c} \times \mathrm{m} + \frac{1}{\mathrm{N_a}}, \tag{5}$$

where N_{c} is sum of the weights $D_t(i)$ of training images classified correctly, N_{a} is the number of active nodes, and m is the number of training images. If the classification performance is the same, the number of active nodes should be small in this fitness function. We describe the features of the training images used in each experiment as follows:

Experiment 1: Texture Images We use texture images from the publicly available database VisTex.[1] We use six classes in this experiment. We make 128 images with 64×64 pixels each by dividing two texture images of 512×512 pixels for each class. The number of training images is 60 (10 images for each class). The main feature of these images is that the test images are comparatively

[1] http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html

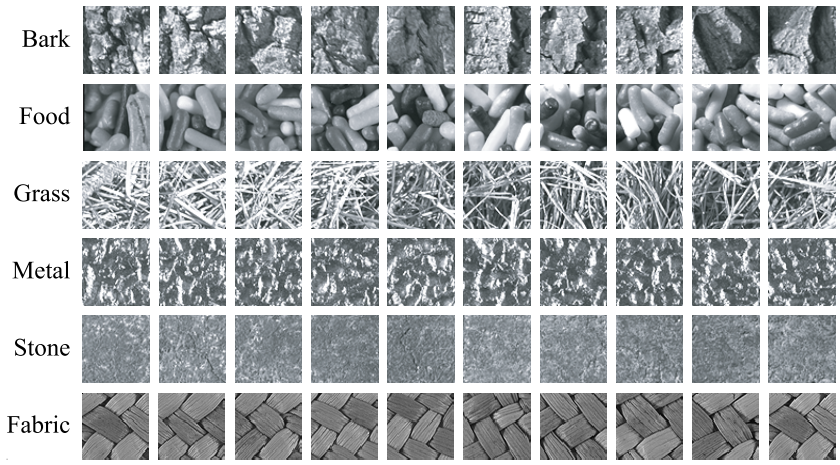

Fig. 3. Training images used in experiment 1 (texture).

Fig. 4. Example images used in experiment 2 (pedestrian).

similar to the training images. The training images used in this experiment are displayed in Fig. 3.

Experiment 2: Pedestrian Images We use 924 pedestrian images from the publicly available database MIT Pedestrian Database[2] and 200 nonpedestrian images. We use two classes in this experiment. The size of all images is 64×128 pixels, and the number of the training images is 200 (100 images for each class). The pedestrian images have various resolutions while the pedestrians are roughly the same size. The nonpedestrian images are manually cut out from outdoor images. An example of the training images used in this experiment is displayed in Fig. 4.

Experiment 3: Generic Object Images We use 500 generic object images from the publicly available WANG image database.[3] We use five classes in this experiment. The size of all images is 96×64 or 64×96 pixels, and the number of

[2] http://cbcl.mit.edu/software-datasets/PedestrianData.html
[3] http://wang.ist.psu.edu/docs/related

Fig. 5. Training images used in experiment 3 (generic object).

Table 2. Discrimination rate for the training and test images (texture).

	Training set		Test set	
Class	GIN-IC	Proposal	GIN-IC	Proposal
	Average	Average	Average ± SD	Average ± SD
Bark	99.0%	100.0%	70.7 ± 9.2%	91.7 ± 4.0%
Food	100.0%	100.0%	86.1 ± 5.8%	98.4 ± 1.8%
Grass	100.0%	100.0%	88.1 ± 3.2%	85.6 ± 6.5%
Metal	90.0%	100.0%	76.7 ± 16.2%	98.3 ± 1.9%
Stone	99.0%	100.0%	80.4 ± 10.5%	94.9 ± 5.7%
Fabric	100.0%	100.0%	94.7 ± 6.1%	99.4 ± 0.8%
Total	98.0%	100.0%	82.8 ± 8.2%	94.7 ± 1.5%

training images is 50 (10 images for each class). The main feature of these images is that target objects have various sizes, positions, types, and so on. Therefore, this problem is more difficult than the other two problems. The training images used in this experiment are displayed in Fig. 5.

5 Results and Discussion

5.1 Results

We compare our proposed method with GIN-IC in discrimination rate for training and test images and time required for learning in this section. The results are the average and the standard deviation (SD) over 10 different runs.

As the result of experiment 1, discrimination rates for the training and test images of textures are shown in Table 2. Our proposed method achieved 100% classification accuracy for all training runs against 98% on an average in GIN-IC.

Table 3. Discrimination rate for the training and test images (pedestrian).

	Training set		Test set	
Class	GIN-IC	Proposal	GIN-IC	Proposal
	Average	Average	Average ± SD	Average ± SD
Pedestrian	89.5%	100.0%	81.4 ± 7.0%	88.6 ± 4.2%
Non-pedestrian	90.5%	100.0%	73.0 ± 10.5%	75.7 ± 4.0%
Total	90.1%	100.0%	80.5 ± 6.1%	87.2 ± 3.7%

Table 4. Discrimination rate for the training and test images (generic object).

	Training set		Test set	
Class	GIN-IC	Proposal	GIN-IC	Proposal
	Average	Average	Average ± SD	Average ± SD
Building	76.0%	100.0%	34.3 ± 19.5%	62.7 ± 6.9%
Bus	95.0%	100.0%	67.6 ± 12.4%	74.8 ± 7.8%
Elephant	95.0%	100.0%	66.3 ± 10.7%	84.0 ± 5.9%
Flower	100.0%	100.0%	70.9 ± 9.1%	86.4 ± 7.3%
Horse	96.0%	100.0%	56.6 ± 19.5%	73.4 ± 7.9%
Total	92.4%	100.0%	59.1 ± 6.3%	76.3 ± 3.5%

Moreover, the proposed method is about three times faster than GIN-IC, for the training images. For test images, the proposed method also obtained about 12% higher classification accuracy as compared to that in GIN-IC totally.

As the result of experiment 2, Table 3 shows discrimination rates for the training and test images of pedestrians. As for the results of experiment 1, the proposed method achieved 100% accuracy for all training runs against 90% on an average in GIN-IC. Moreover, the proposed method is about 50 times faster than GIN-IC, for the training images. Our proposed method also obtained higher classification accuracy for test images as compared to GIN-IC totally.

As the result of experiment 3, discrimination rates for the training and test images of generic objects are shown in Table 4. Similar to the results of experiments 1 and 2, the proposed method achieved 100% classification accuracy for all training runs against about 92% on an average in GIN-IC. Moreover, the proposed method is about two times faster than GIN-IC, for the training images. The proposed method also obtained about 20% higher classification accuracy than GIN-IC totally, for test images.

5.2 Discussion

Our proposed method classified all training images completely in all runs and tends to prevent over fitting as compared to single GIN-IC in these experiments. We confirmed that GIN-IC and AdaBoost go well together. Since we use GIN-IC as weak classifiers, our proposed method can generate and select adequate

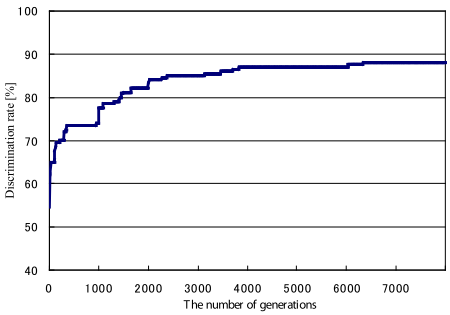

Fig. 6. Example of discrimination rate transition in GIN-IC.

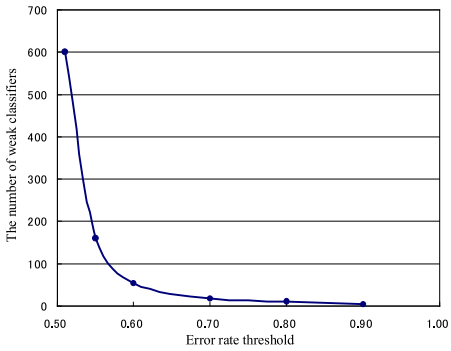

Fig. 7. Relationship between the number of weak classifiers and the error rate threshold τ.

image features by a combination of nodes. Although we only use simple image processing filters and image features as nodes in these experiments, we think that the performance is improved by adding more complex and effective processes such as SIFT descriptor [11]. We should investigate how GIN-IC contributes the performance of our proposed method compared with low level features.

Moreover, our proposed method took lesser time than single GIN-IC, for learning. We attribute this superiority to using GIN-IC as weak classifiers. An example of discrimination rate transition in experiment 2 (pedestrian) is shown in Fig. 6. About 60% classification accuracy is achieved in dozens of generations and higher accuracy costs hundreds and thousands generations. This graph indicates that error rate threshold τ should be small to reduce the number of generations to construct weak classifiers. However, too small τ brings a large number of weak classifiers. The relationship between the number of weak classifiers and τ in pedestrian datasets is shown in Fig. 7. The vertical axis indicates the number of weak classifiers needed to completely classify the training images, and the horizontal axis indicates τ. This graph indicates that many weak classifiers are required if τ is small. From this preliminary experiment, we decide the parameters of τ and T.

6 Conclusions and Future Work

In this paper, we propose a method for automatic construction of an image classifier that aggregates GIN-IC as weak classifiers based on the AdaBoost algorithm. We applied the proposed method to three different problems of image classification, and confirmed that it obtained the optimum solution. In our proposed method, discrimination rates for the training and test images improved in the experiments as compared to that in the previous method GIN-IC. Also, our proposed method reduces the time required for learning.

However, the classifier obtained by the proposed method is constructed with 2500 nodes under 100 weak classifiers while single GIN-IC is constructed with 30 nodes. We will analyze the process of obtaining classifiers and what kind of preprocessing and features were evolved by our proposed method in future. Moreover, this method should be compared with other classifiers to evaluate its quality. Finally, we will apply it to other problems of image classification and object recognition with large scale variation and so on.

References

1. S. Shirakawa, S. Nakayama, and T. Nagao: Genetic Image Network for Image Classification, 11th European Workshop on Evolutionary Computation in Image Analysis and Signal Processing (EvoIASP 2009), Vol. 5484 of LNCS, pp. 395-404, 2009.
2. Y. Freund and R.E. Schapire: Experiments with a New Boosting Algorithm, Proceedings of the 13th International Conference on Machine Leaning (ICML-96), pp. 148-156, 1996.
3. P. Viola and M. Jones: Rapid Object Detection using a Boosted Cascade of Simple Features, Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR 2001), Vol. 1, pp. 511-518, 2001.
4. H. Iba: Bagging, Boosting, and Bloating in Genetic Programming, Proceedings of the Genetic and Evolutionary Computation Conference 1999 (GECCO 1999), Vol. 2, pp. 1053-1060, 1999.
5. G. Folino, C. Pizzuti, and G. Spezzano: Boosting Technique for Combining Cellular GP Classifiers, Proceedings of the 7th European Conference on Genetic Programming (EuroGP 2004), Vol. 3003 of LNCS, pp. 47-56, 2004.
6. G. Folino, C. Pizzuti, and G. Spezzano: GP Ensembles for Large-scale Data Classification, IEEE Transaction on Evolutionary Computation, Vol. 10, No. 5, pp. 604-616, 2006.
7. A. W. Mohemmed, M. Zhang, M. Johnston: Particle Swarm Optimization Based Adaboost for Face Detection, Proceedings of the 2009 IEEE Congress on Evolutionary Computation, IEEE Press, pp. 2494-2501, 2009.
8. H. Schwenk and Y. Bengio: Boosting Neural Networks, Neural Computation, vol. 12, No. 8, pp. 1869-1887, 2000.
9. S. Shirakawa and T. Nagao: Feed Forward Genetic Image Network: Toward Efficient Automatic Construction of Image Processing Algorithm, Advances in Visual Computing: Proceedings of the 3rd International Symposium on Visual Computing (ISVC 2007) Part II, Vol. 4842 of LNCS, pp. 287-297, 2007.
10. J. F. Miller and P. Thomson: Cartesian Genetic Programming, Proceedings of the 3rd European Conference on Genetic Programming (EuroGP 2000), Vol. 1802 of LNCS, pp. 121-132, 2000.
11. D. G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision (IJCV), Vol. 60, No.2, pp. 91-110, 2004.