Involving New Local Search in Hybrid Genetic Algorithm for Feature Selection

Md. Monirul Kabir[1], Md. Shahjahan[2], and Kazuyuki Murase[1]

[1] Department of Human and Artificial Intelligence Systems
University of Fukui, Bunkyo 3-9-1, Fukui 910-8507, Japan
{kabir,murase}@synapse.his.fukui-u.ac.jp
[2] Department of Electrical and Electronic Engineering
Khulna University of Engineering and Technology, Khulna 9203, Bangladesh
jahan@mail.kuet.ac.bd

Abstract. This paper presents a new hybrid genetic algorithm (HGA) for feature selection (FS) called as HGAFS. HGAFS incorporates a new local search operation that is devised and embedded in HGA to fine-tune the search in FS. The proposed local search operation works on basis of the distinct and informative nature of input features that is computed by their correlation information. The aim of using correlation information is to encourage the local search strategy for selecting less correlated (distinct) features. Such an encouragement reduces the redundancy of information in the generated subset of salient features. We have tested our methods on several real-world datasets and have compared the performances with the results of other existing algorithms. It is found that HGAFS produces consistently better performances.

Keywords: Feature selection, local improvement, correlation information, fitness value.

1 Introduction

Feature selection (FS) is a task of reducing the spurious features from the original feature set of a given dataset. Such reduction process ultimately provides the better classification performance in the pattern recognition field and generates a subset of reduced number of salient features. It is known that FS is basically a search process. During searching the spurious features, the approach that depends on the classifier performance as the evaluation function in every FS steps, is called as the wrapper approach [1] while the approach neglecting such evaluation function, is called as filter approach [2]. Furthermore, depending on the selection strategy, searching can be categorized into two ways: forward search [3], and backward search [4]. Apart from these, there are also some other techniques in FS, which are: ant colony optimization (ACO) based search [5] [6], Tabu search, Simulated annealing, and so on.

Genetic search is a recent development guided by genetic algorithm (GA). The GA is biologically inspired and has many mechanisms mimicking natural evolution [7]. It has a lot of prospects in science and engineering optimization or in search problems. Furthermore, GA can be applicable to FS while the problem has an exponential search

C.S. Leung, M. Lee, and J.H. Chan (Eds.): ICONIP 2009, Part II, LNCS 5864, pp. 150–158, 2009.

space. Though GA or, simple GA (SGA) works well in an exponential search space, it suffers by some difficulties such as: inferior solutions caused by premature convergence and poor ability of a fine-tuning near local optimum points [8] [9]. However, to get the better solutions in SGAs, integrating the domain-specific knowledge into SGA, called hybridizing SGA is necessary.

In solving the FS task, there are some hybrid GAs (HGAs) (e.g., [8], [9]) where different types of strategies have been introduced in their local search operations. In [8], the incorporated strategy *ripple*(r) operation in which $2r$-1 times operations are necessary to add the significant features or to delete the insignificant features in (or, from) the selected subset. The computation of significant features here is based on the trained classifier which shows the expensive computational cost. On the other hand, measurement of mutual information (MI) between each pair of features is the main adopted technique in [9] which is also suffered by expensive computation.

As an alternative, this paper proposes a new local search operation in HGA for FS, called as HGAFS that is based on the observation of feature space. In this regard, computation of correlation information is performed to find the relationships between features so that HGAFS can select the distinct and informative features for the pattern classification. The goal of using correlation information is to encourage the search strategy for selecting less correlated (distinct) features. Such encouragement ultimately reduces the redundancy of information in the generated subsets.

A restricted random scheme is also proposed in HGAFS to decide the number of 1-bits (i.e., number of selected features) in individual chromosome of the population set. Such scheme encourages deciding the number of 1-bits in a reduced form. In FS task, it is reasonable in the sense that reduced number of 1-bits ultimately decides the reduced size of subsets. Finally, it can be said that the proposed new local search strategy provides the faster convergence and has an ability to generate reduced subsets of salient features by using its fine-tuning search capability.

The rest of this paper is organized as follows. In Section 2, details of proposed HGAFS are discussed. Experimental results and comparison to other algorithms are reported in Section 3. A short conclusion with few remarks is given in Section 4.

2 Proposed Hybrid GAs for Feature Selection

In this paper, the proposed HGAFS consists of two new techniques which are: (a) a random scheme for deciding the number of 1-bits in the chromosome distributions, (b) a new local search operation that improves the quality of newly generated *offsprings*. The above two contributions put impact positively on the final outcomes of HGAFS. Now, the pseudo-code of the local search operation proposed in HGAFS, which can be applicable for a single *offspring* at a time, is outlined as follows,

```
local-improvement(offsp)  /* offsp: a offspring */
{
  put features of 1-bits in offsp into X;
  compare X with D and S groups; /* D: dissimilar group ; S: similar group */
  divide X into X_d and X_s accordingly;
  switch{
        case |X_d| < δ and |X_s| > ξ : add(δ − |X_d|); rem(|X_s| − ξ)
```

```
        case |X_d| > δ and |X_s| < ξ : rem(|X_d| − δ); add(ξ − |X_s|)
        /* for better understanding see Section 2.2 */
    }
    set 1-bits of X_d and X_s in X accordingly;
    set features of 1-bits in X to offsp;
    set rest of bits in offsp to be 0;
}
```

In HGAFS, there are some fundamental steps which can be explained as follows,

Step 1) Initialize a feature set N of n features, a subset K of k salient features, and a population set P of c chromosomes. Encode the each string of the chromosome set by binary digits representing the value 1 and 0. The value 1 and 0 represent a feature selected and not selected, respectively. Decide the number of 1-bits in each c according to the value of k which can be determined by following Section 2.1.

Step 2) Measure the fitness value of chromosome c in P sequentially using feed-forward three layered neural network (NN) training classifier which can be expressed as,

$$\gamma(Chrom_C) = 100 * (1 - TER) \tag{1}$$

Here, *TER* refers to the testing error rate of the NN on the testing dataset.

Step 3) Perform the conventional 1-point crossover operation [11] by using the conventional rank-based selection procedure [11]. In this case, follow the crossover probability which is defined by user previously.

Step 4) Perform mutation operation according to the conventional scheme [11] over the whole chromosome set in P by following the earlier user defined mutation probability.

Step 5) Perform the local-improvement operation upon one generated *offspring* according to the proposed strategy described in Section 2.2. Repeat the same operation until all generated *offsprings* are covered.

Step 6) Replace the chromosomes of lowest rank order in P by the new local improved *offsprings.*

Step 7) Check the current generation whether it is equal to the predefined total number of generation T, then continue. Otherwise, Go to Step 2).

Step 8) In order to locate the best chromosome that signifies the desired subset of salient features, follow the same procedure that mentioned in [13]. In this case, we select the best chromosome of each generation which is compared with the best chromosome of the previous generation.

HGAFS uses the constructive NN training classifier to compute the fitness of the individual chromosomes. Constructive strategy tries to adjust the suitable number hidden neurons in the hidden layer during training that enhances the classification capability of NN as well. The details description of such strategy can be found in [10]. The following section gives more details about the different components of our proposed algorithm.

2.1 Determination of Subset Size

In HGAFS, deciding the size of salient feature subset fully depends upon the number of 1-bits (k) in the final best chromosome. It should be noted that the value of k in each chromosome once is decided must be fixed up to the final state of FS process. However, if the value of k in each chromosome is too high or, too low then the fitness value may degrade. Thus, by considering the both issues we propose a modified scheme from [13], called *restricted random scheme*. The aim of such scheme is to maintain the value of k in a reasonable range while designing the chromosome set which can be described by two ways:

Firstly, the probabilistic value of k in a bounded region can be defined as,

$$P(k) = \frac{n-k}{\sum_{i=1}^{l}(n-i)} \qquad where,\ 2 \le k \le \psi\ and\ l \in n-k \tag{2}$$

Here, ψ is define by ε of n. Specifically, ε is a user specified parameter and its value is set here up to [0.15, 0.4] depending on the number of n of different datasets. The reason is that, if $\psi \approx n$ then the search space for finding the salient feature subset becomes larger which may cause the high computational cost as well as the ineffective subsets.

Secondly, arrange all the possible values of $P(k)$ to the conventional "roulette-wheel selection" scheme [11] to achieve the value of k consistently.

However, in HGAFS, generating the subset is actually maintained by a predetermined range in between 2 to 12 for its size. On basis of this assumption, the value of ε is determined.

2.2 Local Improvement Operation

In HGAFS, the local improvement operation requires the computation of correlation information of features to find the relationships between features. Therefore, HGAFS can detect the distinct and informative features easily. The following four steps describe the proposed local improvement operation.

i) Measure the correlation (degree of relationships) between different features of a given training set using the well-known *Pearson product-moment correlation coefficient* computation. The correlation coefficient r_{ij} between two features i and j is,

$$r_{ij} = \frac{\sum_{p}(x_i - \bar{x}_i)(x_j - \bar{x}_j)}{\sqrt{\sum_{p}(x_i - \bar{x}_i)^2}\sqrt{\sum_{p}(x_j - \bar{x}_j)^2}} \tag{3}$$

where x_i and x_j are the value of features i and j, respectively. The variables $\bar{x}_i$ and $\bar{x}_j$ represent the mean values of x_i and x_j, averaged over p examples. After computing r_{ij} for all possible combinations of features, HGAFS attempts to compute the correlation of each feature i as,

$$Cor_i = \frac{\sum_{j=1}^{n} r_{ij}}{n-1} \qquad if \quad i \neq j \tag{4}$$

Thereafter, all features n are arranged by ascending order according to their correlation values.

ii) HGAFS then creates two groups. One group contains the first *n/2* features, called as *dissimilar* (*D*) *group*, while the other group contains the remaining *n*/2 features, called as *similar* (*S*) *group*. Now, the first feature of both groups is the least correlated (most distinct) among the other features of each individual group.

iii) Perform the local operation upon the newly generated *offspring*. During this operation, distinguish the number of 1-bit in one set *X*. Then, compare each element of *X* with the element of *D* and *S*. Thus, two subsets X_d and X_s are ultimately formed where those contains the features of *D* and *S*, respectively.

iv) Decide that, X_d and X_s always keep a number of features, say, δ and ξ, respectively. Here, δ is equal to μ of k whereas ξ is to be (1-μ) of k. For this, it is necessary to adjust the above quantities in every step. Since, our motivation is to provide more distinct features to the *offspring* set, therefore, μ here is set to 0.65. However, compare the current $|X_d|$ and $|X_d|$ with δ and ξ, respectively to be made the following two decisions,

(a) $if \ \ |X_d| < \delta \quad add(\delta - |X_d|); \quad if \ \ |X_s| > \xi \quad rem(|X_s| - \xi)$

(b) $if \ \ |X_d| > \delta \quad rem(|X_d| - \delta); \quad if \ \ |X_s| < \xi \quad add(\xi - |X_s|)$

Here, *add* and *rem* operations are performed according to the distinctness of features from *D* and *S* groups. Specifically, *add*() indicates to add more distinct features comparing to the current ones in X_d or X_s whereas *rem*() specifies to be removed more similar (or, less distinct) features comparing to the present ones in X_d or X_s.

3 Experimental Setup

HGAFS was applied to four real-world benchmark datasets to evaluate its performance. The datasets are: breast cancer (BCR), glass (GLS), vehicle (VCL), and sonar (SNR) and the details description of these datasets can be found in [14]. The characteristics of the datasets and their partitions are shown in Table 1. Each experiment was carried out 20 times and the presented results are the average of these 20 runs. The performance of HGAFS was evaluated in terms of the number of selected features (n_s) as well as classification accuracy (*CA*). All experiments were done in Pentium-core 2 duo, 2.66 GHz personal computer.

In this study, we used a number of user specified parameters and their values are decided in some certain ranges. For example, (a) population size=[20,40], (b) crossover probability=0.06, (c) mutation probability=[0.03,0.05], and (d) generation=[20,40]. There are also some other parameters used in NN training while the string of each chromosomes is evaluated. The initial connection weights for an NN were randomly set to [-1.0, 1.0]. The learning rate and momentum term were set to [0.1, 0.2] and [0.5, 0.9], respectively. The number of partial training epochs of NN was chosen between [10, 70]. The NN training was conducted by the well-known BP

algorithm [12]. We conducted one additional set of experiments to investigate the performance of the original all features using constructive NN training classifier.

In course of measuring the fitness value of individual chromosomes, the total examples of the respective dataset were partitioned into three sets. The first 50% examples was selected as a training set to train the NN, the second 25% examples as the validation set to check the condition during training, and the last 25% as the testing set to test the NN.

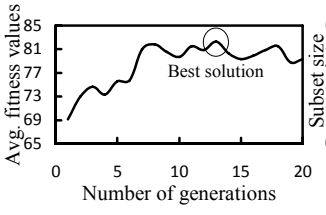

Fig. 1. Generation process of glass dataset for a single run

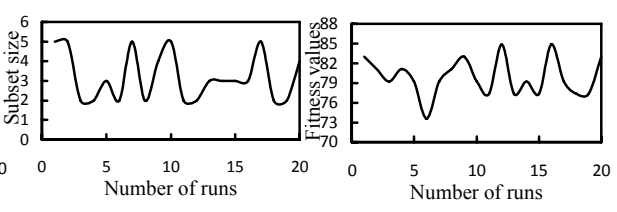

Fig. 2. Variation between subset sizes and fitness values in different runs

3.1 Experimental Results

Table 2 shows the average results of using all features and of selected features (n_s) by HGAFS. The classification accuracy (*CA*) in the table refers to the percentage of correct classification by the trained NN on the testing set. It is seen that a small number of features from the original feature set was selected by HGAFS. For instance, in case of sonar dataset, HGAFS selected 5.35 features on average from a set of 60 features. This indicates that HGAFS could find a reduced number of salient features. The positive effect of a small number of features can be seen when we look the *CA*. For example, the vehicle dataset, the *CA* of all features was 58.77%, when it was 75.79% with 4.40 features. HGAFS also exhibits good results for other datasets.

Furthermore, the use of n_s causes a small standard deviation (*SD*) as presented in the Table 2 for each entry. The low *SDs* refer to the robustness of HGAFS which is indeed the consistency of an algorithm under different initial conditions.

In order to observe how the generation process of HGAFS progresses, Fig. 1 shows the whole scenery of glass problem for a single run. It is seen that average fitness value of the population is being varied while the generation increases. The circle indicates that the maximum average fitness was achieved at that point. The complete information of that point is: the generation number is 13, subset size is 3, and the *CA* of that subset is 83.02%. In contrast, Fig. 2 exhibits the variation curves between the subset size and *CAs* in total 20 runs. Thus, it can be assumed that the performance of a subset is roughly dependent on its size.

3.2 Comparison with Other Works

In this context, the obtained results of HGAFS on four datasets are compared with the results of different FS algorithms. Three well-known algorithms $HGAFS_H$[9], GPFS[13], and ICFS[3] are chosen for comparison. The results are summarized in

Table 3. Since, the different algorithms were evaluated in different experimental setups; therefore, we cannot compare the results of HGAFS completely with other algorithms until the all experiments are performed in the same experimental setup.

Table 1. Characteristics of datasets

Datasets	Features	Classes	Examples	Partition sets		
				Training	Validation	Testing
BCR	9	2	699	349	175	175
GLS	9	6	214	108	53	53
VCL	18	4	846	424	211	211
SNR	60	2	208	104	52	52

Table 2. Average results of BCR, GLS, VCL, and SNR datasets. SD refers standard deviation

Datasets	Avg. results with all features		Avg. results with selected features			
	CA (%)	SD	No. of feature	SD	CA (%)	SD
BCR	97.60	0.002	3.25	1.17	98.55	0.006
GLS	71.51	0.046	3.45	1.07	81.04	0.021
VCL	58.77	0.152	4.40	1.15	75.79	0.009
SNR	70.87	0.092	5.35	2.21	85.87	0.037

Table 3. Comparisons between HGAFS, $HGAFS_H$[9], GPFS[13], and ICFS[3]

Datasets		Comparisons			
		HGAFS	$HGAFS_H$	GPFS	ICFS
BCR	No. of features	3.25	--	2.23	5.00
	Class. acc. (%)	98.55	--	96.84	98.25
GLS	No. of features	3.45	5.00	--	4.50
	Class. acc. (%)	81.04	65.51	--	65.19
VCL	No. of features	4.40	11.00	5.37	--
	Class. acc. (%)	75.79	76.36	78.45	--
SNR	No. of features	5.35	15.00	9.45	--
	Class. acc. (%)	85.87	87.02	86.26	--

Table 3 shows the comparisons between HGAFS and other algorithms on basis of average percentage of *CAs* and average number of n_s. Now, the comparative studies in between HGAFS and other algorithms for four datasets are stated below.

Cancer: The number of n_s in HGAFS is lower than ICFS but comparable with GPFS. In contrast, the overall performance is better than the others.

Glass: HGAFS outperforms $HGAFS_H$ and ICFS significantly in every event. HGAFS drastically reduces the original feature set and optimally generates a reduced number of salient feature subset resulting better *CAs*.

Vehicle: In terms of number of n_s, HGAFS achieved a few number of features comparing to $HGAFS_H$ and GPFS while in case of *CA*, it is comparable to those algorithms.

Sonar: HGAFS achieved a few number of n_s comparing to $HGAFS_H$ and GPFS. But in case of *CA*, it is reasonable or comparable to those algorithms.

4 Conclusion

This paper proposes a new local search approach in HGA for FS that is based on the observation of feature space. Computation of correlation information of features was used to recognize the distinct and informative features which were utilized later in the local search operation of HGAFS. Thus, the proposed local search operation helps to reduce the redundancy of information in the generated subset. In contrast, a restricted random scheme was incorporated to decide the number of 1-bits in individual chromosome of a population set which is reasonable in the sense that reduced number of one decides the reduced size of subsets.

In HGAFS, neither the computation of training based classifier nor the computation of mutual information between a pair of features is taken into account for its local search operation. Apart from these both issues HGAFS achieves a faster convergence and fine-tuning in local optimum points during FS.

Extensive experiments have been carried out in this paper to evaluate how well HGAFS performed on different real-world datasets in comparison with other prominent FS algorithms. In almost all except some few cases, HGAFS outperformed the others in terms of the number of selected features and classification performances. The results of the low *SDs* of the *CAs* exhibit the robustness of this algorithm.

Since the focus of this paper is to present the fundamental ideas and technical details of HGAFS, the detailed comparisons with other algorithms using rigorous statistical methods are left as the future work.

Acknowledgements

Supported by grants to KM from the Japanese Society for Promotion of Sciences, the Yazaki Memorial Foundation for Science and Technology, and the University of Fukui.

References

1. Hsu, C., Huang, H., Schuschel, D.: The ANNIGMA-wrapper approach to fast feature selection for neural nets. IEEE Trans. on Syst., Man, and Cybern.-Part B: Cybern. 32(2), 207–212 (2002)
2. Hall, M.A.: Correlation-based feature selection for discrete and numeric class machine learning. In: 17th International Conference on Machine Learning (2000)
3. Guan, S., Liu, J., Qi, Y.: An incremental approach to contribution-based feature selection. Journal of Intelligence Systems 13(1) (2004)

4. Abe, S.: Modified backward feature selection by cross validation. In: Proceedings of the European Symposium on Artificial Neural Networks, pp. 163–168 (2005)
5. Aghdam, M.H., Aghaee, N.G., Basiri, M.E.: Text feature selection using ant colony optimization. Expert systems with applicantions 36, 6843–6853 (2009)
6. Ke, L., Feng, Z., Ren, Z.: An efficient ant colony optimization approach to attribute reduction in rough set theory. Pattern Recognition Letters 29, 1351–1357 (2008)
7. Holland, J.: Adaptation in Nature and Artificial Systems. MIT Press, Cambridge (1992)
8. Oh, I.-S., Lee, J.S., Moon, B.: Hybrid Genetic Algorithms for Feature Selection. IEEE Trans. on Pattern Analysis and Machine Intelligence 26(11), 1424–1437 (2004)
9. Huang, J., Cai, Y., Xu, X.: A hybrid genetic algorithm for feature selection wrapper based on mutual information. Pattern Recognition Letters 28, 1825–1844 (2007)
10. Kwok, T.Y., Yeung, D.Y.: Objective functions for training new hidden units in constructive neural networks. IEEE Trans. Neural Network 8(5), 1131–1148 (1997)
11. Goldberg, D.E.: Genetic Algorithms in search, optimization and machine learning (2004)
12. Rumelhart, D.E., McClelland, J.: Parallel distributed processing. MIT Press, Cambridge (1986)
13. Muni, D.P., Pal, N.R., Das, J.: Genetic Programming for Simultaneous Feature Selection and Classifier Design. IEEE Trans. on Systems, Man, and Cybern.-Part B: Cybern. 36(1) (2006)
14. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases. Dept. of Information and Computer Sciences, University of California, Irvine (1998), http://www.ics.uci.edu/~mlearn/MLRepository.html