An Improved Genetic Algorithm and Its Application in Classification

Xuesong Yan[1,2], Wenjing Luo[1], Wei Li[1], Wei Chen[1], Can Zhang[1] and Hanmin Liu[3]

**[1] School of Computer Science, China University of Geosciences
Wuhan, Hubei 430074, China**

**[2] Department of Computer Science, University of Central Arkansas
Conway, AR 72035, USA**

**[3] Wuhan Institute of Ship Building Technology
Wuhan, Hubei 430050, China**

Abstract

In this paper, based on a simple genetic algorithm and combine the base ideology of orthogonal design method then applied it to the population initialization, using the intergenerational elite mechanism, as well as the introduction of adaptive local search operator to prevent trapped into the local minimum and improve the convergence speed to form a new genetic algorithm. Through the series of numerical experiments, the new algorithm has been proved to be efficiency. we also use this new algorithm in data classification, select 5 benchmark datasets and the experiment results shown the new algorithm can get higher accuracy than k-nearest neighbor method.

Keywords: *Genetic Algorithm, Optimization, Classification, K-Nearest Neighbor, Population.*

1. Introduction

Candidate solutions to some problems are not simply deemed correct or incorrect but are instead rated in terms of quality and finding the candidate solution with the highest quality is known as optimization. Optimization problems arise in many real-world scenarios. Take for example the spreading of manure on a cornfield, where depending on the species of grain, the soil quality, expected amount of rain, sunshine and so on, we wish to find the amount and composition of fertilizer that maximizes the crop, while still being within the bounds imposed by environmental law.

Several challenges arise in optimization. First is the nature of the problem to be optimized which may have several local optima the optimizer can get stuck in, the problem may be discontinuous, candidate solutions may yield different fitness values when evaluated at different times, and there may be constraints as to what candidate solutions are feasible as actual solutions to the real-world problem. Furthermore, the large number of candidate solutions to an optimization problem makes it intractable to consider all candidate solutions in turn, which is the only way to be completely sure that the global optimum has been found. This difficulty grows much worse with increasing dimensionality, which is frequently called the curse of dimensionality, a name that is attributed to Bellman, see for example [1]. This phenomenon can be understood by first considering an n-dimensional binary search-space. Here, adding another dimension to the problem means a doubling of the number of candidate solutions. So the number of candidate solutions grows exponentially with increasing dimensionality. The same principle holds for continuous or real-valued search-spaces, only it is now the volume of the search-space that grows exponentially with increasing dimensionality. In either case it is therefore of great interest to find optimization methods which not only perform well in few dimensions, but do not require an exponential number of fitness evaluations as the dimensionality grows. Preferably such optimization methods have a linear relationship between the dimensionality of the problem and the number of candidate solutions they must evaluate in order to achieve satisfactory results, that is, optimization methods should ideally have linear time-complexity O(n) in the dimensionality n of the problem to be optimized.

Another challenge in optimization arises from how much or how little is known about the problem at hand. For example, if the optimization problem is given by a simple formula then it may be possible to derive the inverse of that formula and thus find its optimum. Other families of problems have had specialized methods developed to optimize them efficiently. But when nothing is known about the optimization problem at hand, then the No Free Lunch (NFL) set of theorems by Wolpert and Macready states that any one optimization method will be as likely as any other to find a satisfactory solution [2]. This is especially important in deciding what performance goals one should have when designing new optimization methods, and whether one should attempt to devise the ultimate optimization method which will adapt to all problems and perform well. According to the NFL theorems such an optimization method does not exist and the focus of this thesis will therefore be on the opposite: Simple optimization methods that perform well for a range of problems of interest.

The most popular evolutionary model used in the current research is Genetic Algorithms (GA), originally developed by John Holland [3]. The GA reproduction operators, such as recombination and mutation, are considered analogous to the biological process of mutation and crossover respectively in population genetics. The recombination operator is traditionally used as the primary search operator in GA while the mutation operator is considered to be a background operator, which is applied with a small probability.

Genetic algorithms have been successfully used in data mining, in order to determine classification rules [4], in order to search for appropriate cluster centers) [5], to select the attributes of interest in predicting the value of a target attribute [6], etc. Classification of instances was performed using some hybrid algorithms based on genetic algorithms and particle swarm optimization [7], respectively Naive Bayes and k-Nearest Neighbors [8]. A few applications in which genetic algorithms were successfully applied to solve classification problems are prints classification, heart disease classification, classification of emotions on the human face, etc.

2. Improved Genetic Algorithm

In general, genetic algorithms are usually used to solve problems with little or no domain knowledge, NP-complete problems, and problems for which near optimum solution is sufficient. The GA methods can be applied only if there exist a reasonable time and space for evolution to take place. But the traditional genetic algorithm has the shortcoming: trapped into the local minimum easily [9].

2.1 Population Initialization

The traditional method of genetic algorithm is randomly initialized population, that is, generate a series of random numbers in the solution space of the question. Design the new algorithm, we using the orthogonal initialization [10] in the initialization phase. For the general condition, before seeking out the optimal solution the location of the global optimal solution is impossible to know, for some high-dimensional and multi-mode functions to optimize, the function itself has a lot of poles, and the global optimum location of the function is unknown. If the initial population of chromosomes can be evenly distributed in the feasible solution space, the algorithm can evenly search in the solution space for the global optimum. Orthogonal initialization is to use the orthogonal table has the dispersion and uniformity comparable; the individual will be initialized uniformly dispersed into the search space, so the orthogonal design method can be used to generate uniformly distributed initial population.

2.2 Elite Select Mechanism

Genetic algorithm is usually complete the selection operation based on the individual's fitness value, in the mechanism of intergenerational elite, the population of the front generation mixed with the new population which generate through crossover and mutation operations, in the mixed population select the optimum individuals according to a certain probability. The specific procedure is as follows:
Step1: using crossover and mutation operations for population P1 which size is N then generating the next generation of sub-populations P2;
Step2: The current population P1 and the next generation of sub-populations P2 mixed together form a temporary population;
Step3: Temporary population according to fitness values in descending order, to retain the best N individuals to form new populations P1.

The characteristic of this mechanism is mainly in the following aspects. First is robust, because of using this selection strategy, even when the crossover and mutation operations to produce more inferior individuals, as the results of the majority of individual residues of the original population, does not cause lower the fitness value of the individual. The second is in genetic diversity maintaining, the operation of large populations, you can better maintain the genetic diversity of the population evolution process. Third is in the sorting method, it is good to overcome proportional to adapt to the calculation of scale.

2.3 Adaptive Search Operator

Local search operator has a strong local search ability, and then can solve the shortcomings of genetic algorithm has the weak ability for the local search. And the population according to the current state of adaptive evolution of the local search space adaptive local search operator will undoubtedly greatly enhance the ability of local search. In the initial stage of the evolution, the current optimal solution from the global optimum region is still relatively far away, this time the adaptive local search operator to require search a large neighborhood space to find more optimal solution, it can maintain the population diversity.

When the population has evolved to the region containing the global optimum, the adaptive local search operator to require a relatively small area to search in order to improve the accuracy of the global optimal solution.

In our algorithm, the adaptive local search operator is the adaptive orthogonal local search operator. Adaptive orthogonal local search operator is aimed at the neighborhood of a point to search, so the key point is to identify a point as the center of the hypercube, the hypercube in the orthogonal test, expect to be better solution.

2.4 Experiment Simulation

We design the experiments to study its convergence speed by comparing with a traditional genetic algorithm (GA). Ten benchmark functions are selected. One of them is a multimodal function, which is a very difficult function (explained in its function description). We choose it because we want to investigate not only their convergence speeds, but also their abilities of finding the optimal solutions. The simple description of each function is given as follows.

F1: Schaffer function

$$\min f(x_i) = 0.5 - \frac{(\sin^2\sqrt{x_1^2+x_2^2} - 0.5)}{[1+0.001(x_1^2+x_2^2)]^2},$$
$$-100 \le x_i \le 100$$

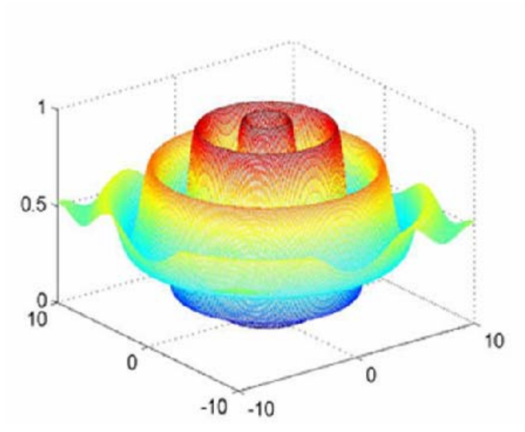

Fig. 1 Schaffer Function.

The global optimal value of this function is 1.0, located at the central point with coordinates (0, 0), and the circle with the radius 3.14 on the overall situation from numerous major points of the uplift. This function has a strong shock. Therefore, it is difficult to find a general method, which can find its global optimal solution.

F2: Shubert function

$$\min f(x,y) = \left\{\sum_{i=1}^{5} i\cos\left[(i+1)x+i\right]\right\}\times$$
$$\left\{\sum_{i=1}^{5} i\cos\left[(i+1)y+i\right]\right\}, x, y \in [-10,10]$$

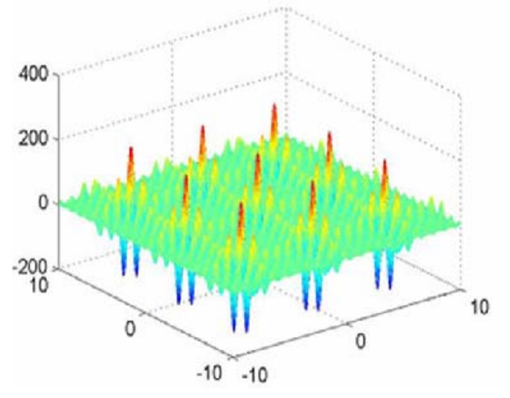

Fig. 2 Shubert Function.

This function has 760 local minima and 18 global ones. The global minimum value is -186.7309.

F3: Hansen function

$$\min f(x,y) = \sum_{i=1}^{5} i\cos((i-1)x+i)$$
$$\sum_{j=1}^{5} j\cos((j+1)y+j), x, y \in [-10,10]$$

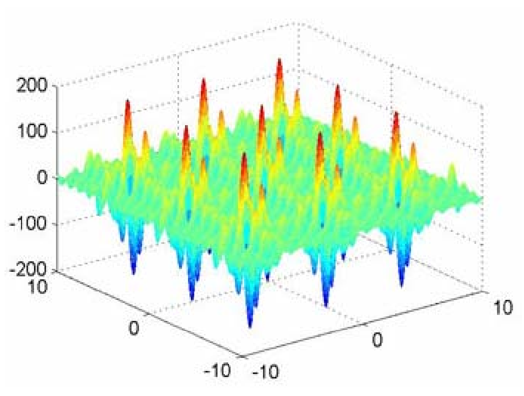

Fig. 3 Hansen Function.

This function has a global minimum value -176.541793, in the following nine points, i.e., (-7.589893, -7.708314), (-7.589893, -1.425128), (-7.589893, 4.858057), (-1.306708, -7.708314), (-1.306708, -1.425128), (-1.306708, 4.858057), (4.976478, -7.708314), (4.976478, -7.708314), and (4.976478, 4.858057). It also has 760 local minima.

F4: Camel function

$$\min f(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + \left(-4 + 4y^2\right)y^2, x, y \in [-100,100]$$

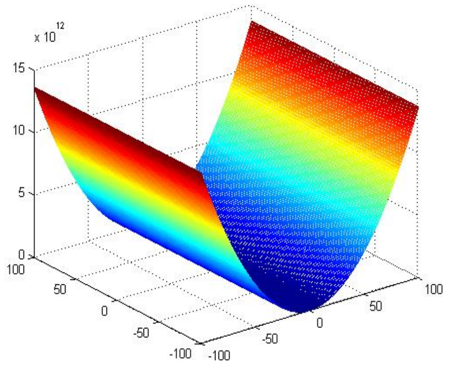

Fig. 4 Camel function.

Camel function has six local minima, i.e., (1.607105, 0.568651), (-1.607105, -0.568651), (1.703607, -0.796084), (-1.703607, 0.796084), (-0.0898, 0.7126) and (0.0898, -0.7126). It has two global minimum points, i.e., (-0.0898, 0.7126) and (0.0898, -0.7126). Its global minimum is -1.031628.

The function 5 (called F5 in this paper) can be stated as follows:

$$\min f(x_1,x_2) = \begin{Bmatrix} 1 + (x_1 + x_2 + 1)^2 \\ (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \end{Bmatrix} \times \begin{Bmatrix} 30 + (2x_1 - 3x_2)^2 \\ (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \end{Bmatrix}, \quad x_1, x_2 \in [-50,50]$$

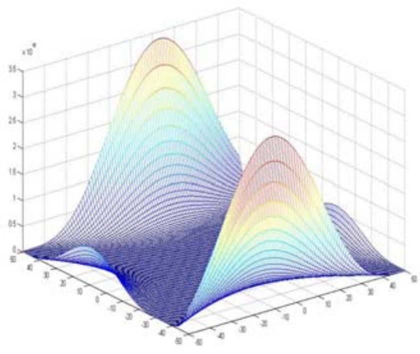

Fig. 5 Function 5.

This function is an eighth-order polynomial with two variables, shown in Figure 5. However, it has four local minima, including a global one, i.e., $f(1.2,0.8) = 840.0$, $f(1.8,0.2) = 84.0$, $f(0.6,0.4) = 30.0$ and $f^*(0,1.0) = 3.0$ (global minimum).

The function 6 (called F6 in this paper) can be stated as follows:

$$\min f(x,y) = -\begin{bmatrix} x\sin(9\pi y) + \\ y\cos(25\pi x) + 20 \end{bmatrix}, \quad x, y \in [-10,10]$$

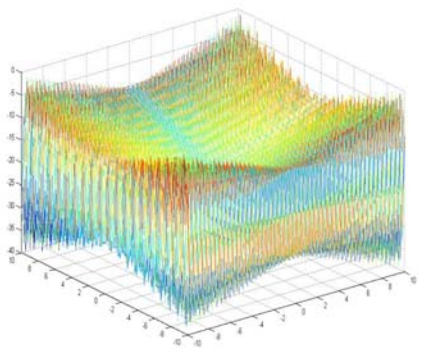

Fig. 6 Function 6.

This is a very complex and difficult function. First, it is a multimodal function. Besides, $\sin(9\pi y)$ and $\cos(25\pi x)$ are high frequency oscillations in the different directions. Furthermore, its peak (or ravine) of the function is intensive at the points when $|x,y| \to 10$. The scene of this function is also very complicated, shown in Figure 6. The global optimal of this function is (-10, 9.9445695) = -39.944506953367.

The function 7 (called F7 in this paper) is defined as follows:

$$\min f(x_1,x_2) = 20 + x_1^2 + x_2^2 - 10 * (\cos 2\pi x_1 + \cos 2\pi x_2)$$

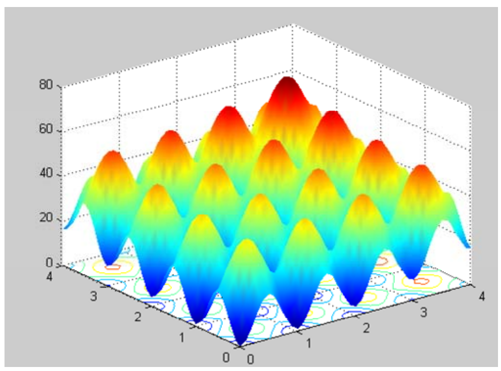

Fig. 7 Function 7.

Its minimum value is 0, as shown in Figure 7.

The function 8 (called F8 in this paper) is defined as follows:

$$\min f(x_1,x_2)=100*(x_1^2-x_2)^2+(1-x_1)^2,$$
$$x\in[-2.048,2.048]$$

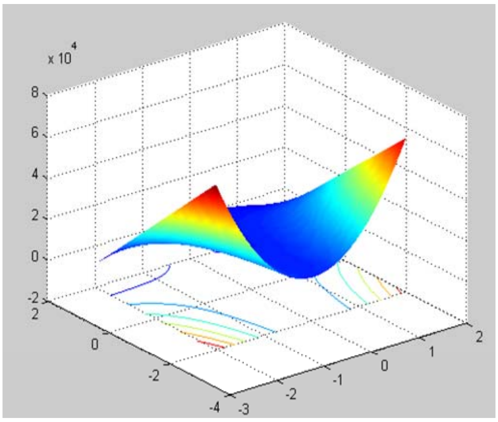

Fig. 8 Function 8.

Its minimum value is 0, as shown in Figure 8.

The function 9 (called F9 in this paper) is defined as follows:

$$\min f(x_1,x_2)=x_1^2+x_2^2, x_1,x_2\in[-100,100]$$

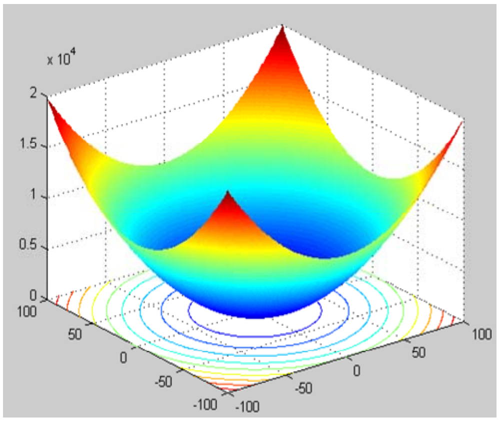

Fig. 9 Function 9.

Its minimum value is 0, as shown in Figure 9.

The function 10 (called F10 in this paper) is defined as follows:

$$\min f(x_1,x_2)=0.5x_1^2+0.5(1-\cos 2x_2)+x_2^2$$

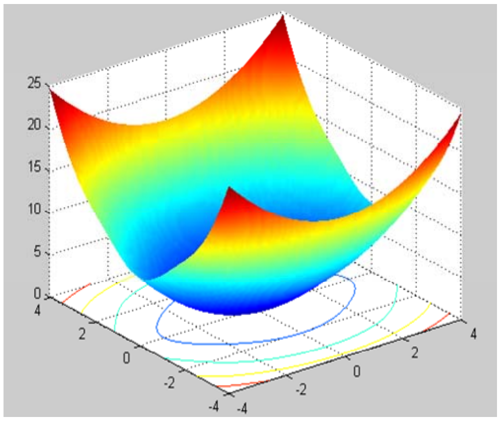

Fig. 10 Function 10.

Its minimum value is 0, as shown in Figure 10.

In order to obtain a solid comparison between GA and improved GA (IGA), we run each algorithm 100 times for the ten functions described above. Our experimental results are shown in Table 1, including the best solution and the number of times of finding the best solution for each function. For example, on the most difficult function F6 among the ten functions, GA could not find its optimal solution (i.e, 0 times out of 100 runs). The best solution GA achieved is -14.786954. However, PSO found its optimal solution (-39.944506) five times out of 100 runs).

Table 1: The number of times of both GA and PSO achieve optimal solutions among 100 runs on the ten functions

Function	***Algorithm***	***Convergence Times***	***Optimal Solution***
F1	GA	72	1.0000000
	IGA	75	1.0000000
F2	GA	75	-186.730909
	IGA	82	-186.730909
F3	GA	85	-176.541793
	IGA	91	-176.541793
F4	GA	23	-1.031628
	IGA	58	-1.031628
F5	GA	16	3.000000
	IGA	25	3.000000
F6	GA	0	-14.786954
	IGA	8	-39.944506
F7	GA	90	0.000000
	IGA	97	0.000000
F8	GA	90	0.000000
	IGA	98	0.000000

F9	GA	100	0.000000
	IGA	100	0.000000
F10	GA	93	0.000000
	IGA	98	0.000000

From Table 1, we can see that IGA achieves optimal solution more frequently than GA does on nine out of the ten functions, except the easiest one (i.e., F9). On the easiest function F9, both of them achieve the optimal solution in all 100 runs. From the experimental results in Table 1, we can conclude that the IGA algorithm has more efficient global searching capability than the GA algorithm. Our experiments verified that IGA converges more quickly than the GA algorithm.

3. K-Nearest Neighbor Classification Algorithm

The nearest neighbor method [11, 12] represents one of the simplest and most intuitive techniques in the field of statistical discrimination. It is a nonparametric method, where a new observation is placed into the class of the observation from the learning set that is closest to the new observation, with respect to the covariates used. The determination of this similarity is based on distance measures.

Formally this simple fact can be described as follows: Let $L=\{(y_i,x_i), i=1,2,...,n_L\}$ be training or learning set of observed data, where $y_i \in \{1,2,...,c\}$ denotes class membership and the vector $x_i^{'}=(x_{i1},x_{i2},...,x_{ip})$ represents the predictor values. The determination of the nearest neighbors is based on an arbitrary distance function $d(.,.)$. Then for a new observation (y,x) the nearest neighbor $(y_{(1)},x_{(1)})$ within the learning set is determined by $d(x,x_{(1)})=\min_i(d(x,x_i))$ and $\hat{y}=y_{(1)}$ the class of the nearest neighbor is selected as prediction for y. The notation $x_{(j)}$ and $y_{(j)}$ here describes the jth nearest neighbor of x and its class membership, respectively.

For example, such typical distance functions are the Euclidean distance $d(x,x_j)=(\sum_{s=1}^{p}(x_{is}-x_{js})^2)^{\frac{1}{2}}$.

The method has been explained by the random occurrence of the learning set, as described in Fahrmeir et al. [13]. The class label $y_{(1)}$ of the nearest neighbor $x_{(1)}$ of a new case x is a random variable. So the classification probability of x into class $y_{(1)}$ is $P(y_{(1)}\mid x_{(1)})$. For large learning sets x and $x_{(1)}$ coincide very closely with each other, so $P(y_{(1)}\mid x_{(1)})\approx P(y\mid x)$ results approximately. Therefore the new observation x is predicted as belonging to the true class y with the probability approximately $P(y\mid x)$.

A first extension of this idea, which is widely and commonly used in practice, is the so-called k-nearest neighbor method. Here not only the closest observation within the learning set is referred for classification, but also the k most similar cases. The parameter k has to be selected by the user. Then the decision is in favor of the class label, most of these neighbors belong to.

Let k_r denote the number of observations from the group of the nearest neighbors, that belong to class r: $\sum_{r=1}^{c}k_r=k$. Then a new observation is predicted into the class l with $k_l=\max_r(k_r)$. This prevents one singular observation from the learning set deciding about the predicted class. The degree of locality of this technique is determined by the parameter k: For $k=1$ one gets the simple nearest neighbor method as maximal local technique, for $k\to n_L$ a global majority vote of the whole learning set results. This implies a constant prediction for all new observations that have to be classified: Always the most frequent class within the learning set is predicted.

K-Nearest Neighbor (KNN) is one of the most popular algorithms for pattern recognition. Many researchers have found that the KNN algorithm accomplishes very good performance in their experiments on different data sets.
The traditional KNN text classification has three limitations [14]:
1. High calculation complexity: To find out the k nearest neighbor samples, all the similarities between the training samples must be calculated. When the number of training samples is less, the KNN classifier is no longer optimal, but if the training set contains a huge number of samples, the KNN classifier needs more time to calculate the similarities. This problem can be solved in 3 ways: reducing the dimensions of the feature space; using smaller data sets; using improved algorithm which can accelerate to [15];
2. Dependency on the training set: The classifier is generated only with the training samples and it does not use any additional data. This makes the algorithm to depend on the training set excessively; it needs recalculation even if there is a small change on training set;
3. No weight difference between samples: All the training samples are treated equally; there is no difference between the samples with small number of data and huge number

of data. So it doesn't match the actual phenomenon where the samples have uneven distribution commonly.

4. Classification Experiment

String Representation [16]-Here the chromosomes are encoded with real numbers; the number of genes in each chromosome represents the samples in the training set. Each gene will have 5 digits for vector index and k number of genes. For example, if k=5, a sample chromosome may look as follows:
00100 10010 00256 01875 00098

Here, the 00098 represents, the 98th instance and the second gene say that the 1875 instance in the training sample. Once the initial population is generated now we are ready to apply genetic operators. With these k neighbors, the distance between each sample in the testing set is calculated and the accuracy is stored as the fitness values of this chromosome.

The algorithm process step is given as Fig. 11.

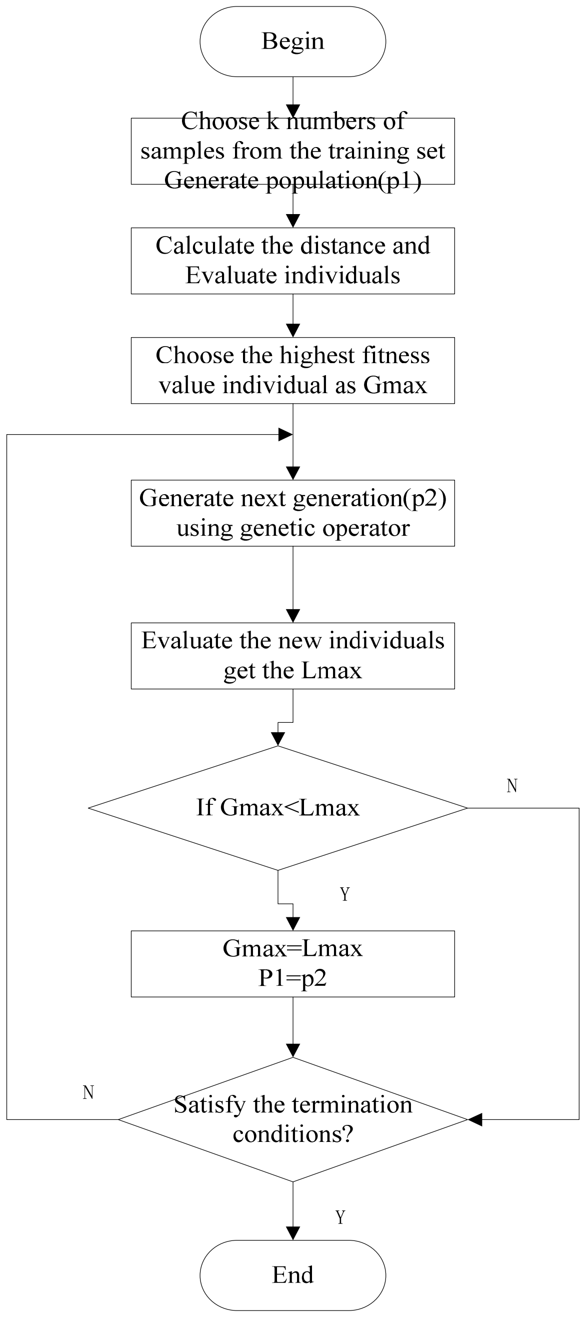
Begin
Choose k numbers of samples from the training set Generate population(p1)
Calculate the distance and Evaluate individuals
Choose the highest fitness value individual as Gmax
Generate next generation(p2) using genetic operator
Evaluate the new individuals get the Lmax
If Gmax<Lmax
N
Y
Gmax=Lmax
P1=p2
N
Satisfy the termination conditions?
Y
End

Fig. 11 Algorithm framework

The performance of the approaches discussed in this paper has been tested with 5 different datasets, downloaded from UCI machine learning data repository. All experiments are performed on Intel Core(TM)2 Duo CPU 2.26GHz/4G RAM Laptop. Each datasets run 10 times with different k values. Table 2 shows the details about the datasets used in this paper.

Table 2: Experiment dataset

Dataset Name	***Total No. of Instances***	***Total No. of Features***
Balance	624	5
Iris	150	4
Sonar	208	60
Glass	214	10
Ionosphere	351	34

Table 3 depicts the performance accuracy of our proposed classifier compared with traditional KNN. From the results it is shown that our proposed method outperforms the traditional KNN method with higher accuracy.

5. Conclusions

This paper introduces a new algorithm based on the traditional genetic algorithm, for the traditional GA algorithm the new algorithm has done some improvements: By introducing genetic selection strategy, decreased the possibility of being trapped into a local optimum. Compared the traditional genetic algorithm, the new algorithm enlarges the searching space and the complexity is not high. By analyzing the testing results of benchmarks functions optimization, we reach the conclusion: in the optimization precision, the new algorithm is efficiency than the traditional genetic algorithm. We also use this new algorithm for data classification and the experiment results shown that our proposed algorithm outperforms the KNN with greater accuracy.

Acknowledgments

This paper is supported by Natural Science Foundation of China. (No.61272470 and No.61203307), National Civil Aerospace Pre-research Project of China, the Provincial Natural Science Foundation of Hubei (2012FFB04101), and the Fundamental Research Founds for National University, China University of Geosciences(Wuhan). The author thanks Dr. Victor Sheng of Department of Computer Science, University of Central Arkansas, U. S. A for his help.

References

[1] R. Bellman, “Dynamic Programming”, Princeton University Press, 1957.
[2] D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization”, IEEE Transactions on Evolutionary Computation, 1(1), 1997, pp.67-82.
[3] J. Holland, “Adaptation in natural and artificial systems”, University of Michigan press, 1975.
[4] S.Dehuri, A. Ghosh and R. Mall, “Genetic Algorithms for Multi-Criterion Classification and Clustering in Data Mining”, International Jurnal of Computing & Information Sciences, 2006, pp. 143-154.
[5] U. Maulik and S. Bandyopadhyay, “Genetic algorithm-based clustering technique”, Pattern Recognition 33, 2000, pp. 1455-1465.
[6] A.A. Freitas, “A survey of evolutionary algorithms for data mining and knowledge discovery”, Advances in Evolutionary Computation, Springer-Verlag, 2002, pp. 819-845.
[7] R. Ding, H. Dong, X. Feng and G. Yin, “A Hybrid Particle Swarm Genetic Algorithm for Classification”, Proceedings of the 2009 International Joint Conference on Computational Sciences and Optimization, vol.2, 2009, pp. 301.
[8] M. Aci, C. Inan and M. Avci, “A hybrid classification method of k nearest neighbor”, Bayesian methods and genetic algorithm, Expert Systems with Applications, Vol. 37, 2010, pp. 5061-5067.
[9] Xuesong Yan, Qinghua Wu, Can Zhang, Wei Li, Wei Chen, Wenjing Luo, “An Improved Genetic Algorithm and Its Application” TELKOMNIKA Indonesian Journal of Electrical Engineering, vol. 10, No. 5, 2012, pp. 1081-1086.
[10] Leung Yiu-Wing, Wang Yuping, “An orthogonal genetic algorithm with quantization for global numerical optimization”, IEEE Transactions on Evolutionary Computation, 5(1), 2001, pp. 41-53.
[11] E. Fix, and J. Hodges, “Discriminatory analysis Nonparametric discrimination: Consistency properties”, Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
[12] T.M. Cover, and P.E. Hart, “Nearest neighbor pattern classification”, IEEE Transactions on Information Theory, 13, pp. 21–27, 1967.
[13] Fahrmeir, Hamerle and Tutz, “Multivariate statistische Verfahren”, Walter de Gruyter & Co Verlag; Berlin, 1996.
[14] W. Yu, and W. Zhengguo, “A fast kNN algorithm for text categorization”, Proceedings of the Sixth International Conference on Machine Learning and Cybernetics, 2007, pp. 3436-3441.
[15] W. Yi, B. Shi, and W. Zhang’ou, “A Fast KNN Algorithm Applied to Web Text Categorization”, Journal of The China Society for Scientific and Technical Information, 26(1), pp. 60-64, 2007.
[16] N. Suguna and Dr. K. Thanushkodi, “An Improved k-Nearest Neighbor Classification Using Genetic Algorithm”, International Journal of Computer Science Issues, Vol. 7, Issue 4, No 2, 2010, pp. 18-21.

Xuesong Yan associate professor received him B.E. degree in Computer Science and Technology in 2000 and M.E. degree in Computer Application from China University of Geosciences in 2003, received he Ph.D. degree in Computer Software and Theory from Wuhan University in 2006. He is currently with School of Computer Science, China University of Geosciences, Wuhan, China and now as a visiting scholar with Department of Computer Science, University of Central Arkansas, Conway, USA. He research interests include evolutionary computation, data mining and computer application.

Wenjing Luo received her B.E. degree in Computer Science and Technology in 2012. She is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Wei Li received her B.E. degree in Computer Science and Technology in 2012. She is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Wei Chen received him B.E. degree in Computer Science and Technology in 2012. He is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Can Zhang received him B.E. degree in Computer Science and Technology in 2011. He is currently is the M.E. degree candidate with School of Computer Science, China University of Geosciences, Wuhan, China. Her research interests include evolutionary computation.

Hanmin Liu associate professor. He is currently as a Ph.D candidate of School of Computer Science, China University of Geosciences, Wuhan, China. He research interests include evolutionary computation and applications.

Table 3: Experiment results comparison

Dataset Name	*K Value*	*Algorithm*	*Best Accuracy*	*Worst Accuracy*	*Mean Accuracy*
Balance	3	IGA	0.904255	0.840426	0.869903
		KNN	0.914894	0.824468	0.866489
	5	IGA	0.882979	0.824468	0.860511
		KNN	0.941489	0.851064	0.875
	7	IGA	0.925532	0.81383	0.863862
		KNN	0.898936	0.819149	0.86117
	9	IGA	0.909574	0.84	0.877272
		KNN	0.920213	0.840426	0.882979
Iris	3	IGA	0.977778	0.866667	0.933333
		KNN	1	0.933333	0.973333
	5	IGA	1	0.933333	0.973867
		KNN	1	0.911111	0.971111
	7	IGA	1	0.911111	0.968889
		KNN	1	0.888888	0.948889
	9	IGA	1	0.96	0.981683
		KNN	1	0.955556	0.977778
Sonar	3	IGA	0.920635	0.825397	0.868173
		KNN	0.904762	0.746032	0.806349
	5	IGA	0.904762	0.714286	0.82618
		KNN	0.857143	0.666667	0.78254
	7	IGA	0.904762	0.714286	0.790363
		KNN	0.888889	0.587302	0.739683
	9	IGA	0.934564	0.698413	0.787446

		KNN	0.873016	0.603175	0.736508
Glass	3	IGA	0.923077	0.643357	0.765542
		KNN	0.861538	0.584615	0.713846
	5	IGA	0.783516	0.676923	0.745714
		KNN	0.784615	0.630769	0.687692
	7	IGA	0.830769	0.630769	0.730769
		KNN	0.753846	0.569231	0.672308
	9	IGA	0.784615	0.553846	0.671057
		KNN	0.753846	0.553846	0.661538
Ionosphere	3	IGA	0.981132	0.871749	0.927245
		KNN	0.943396	0.811321	0.866038
	5	IGA	0.943396	0.867925	0.911003
		KNN	0.915094	0.830189	0.883962
	7	IGA	0.962264	0.792453	0.900066
		KNN	0.896226	0.764151	0.834906
	9	IGA	0.933962	0.839623	0.90321
		KNN	0.858491	0.716981	0.774528