Experimental results. The approach described above has been applied on the preprocessed corpus which has been provided by Speech Cycle company.

We have tested standard classification algorithms (k-nearest neighbors algorithms (k-NN), Bayes classifiers, Decision Stump, Rule Induction, perceptron) and the proposed approach on the database with learning and classification for all classes, with learning and classification only for informative classes, and with learning for all classes and classification only for informative classes. We tested our TRE approach fir two-stage classification problem definition. We use TRE approach with agglomerative hierarchical clustering (it is the best one) for "garbage" class, 4-NN, and 9-NN as classification methods for the first stage or the classification problem.

The results of numerical results you can see in table 2. In this table you can see that our new TRE-approach is the most effective for identification of informative classes. For "garbage" class identification it is better to use k-nearest neighbors algorithms.

Conclusions. This paper reported on call classification experiments on large corpora using a new term relevance estimation approach. We propose to split the classification task into two steps: 1) "garbage" class identification; 2) further classification into meaningful classes. The performance of the proposed algorithm is compared to several standard classification algorithms on the database without the "garbage" class and found to outperform them with the accuracy rate of 85,55 %. Combination of our approach with 9-NN algorithm for two-stage classification problem definition provides the accuracy rate of 77,11 % for test sample at whole.

We can conclude that our approach is appropriate and effective for call routing problem.

References

1. Chu-Carroll J. and Carpenter B. Vector-based natural language call routing, *Comput. Linguist.*, 1999, vol. 25, no. 3, p. 361–388.

2. Lee C.-H., Carpenter B., Chou W., Chu-Carroll J., Reichl W., Saad A., Zhou Q. On natural language call routing, *Speech Commun.*, Aug. 2000, vol. 31, no. 4, p. 309–320.

3. Gorin A. L., Riccardi G., and Wright J. H. How may I help you? *Speech Commun.*, 1997, vol. 23, p. 113–127.

4. Schapire R. E., Singer Y. BoosTexter: a boosting-based system for text categorization, *Mach. Learn.*, 2000, vol. 39, no. 2/3, p. 135–168.

5. Albalate A., Suendermann D., Pieraccini R., Minker W. 2009. *Mathematical Analysis of Evolution, Information, and Complexity*, Wiley, Hoboken, USA.

6. Albalate A., Suendermann D., Minker W. *International Journal on Artificial Intelligence Tools*, 2011, 20(5).

7. Albalate A., Suchindranath A., Suendermann D., Minker W.. 2010. *Proc. of the Interspeech 2010, 11th Annual Conference of the International Speech Communication Association*, Makuhari, Japan.

8. Albalate A., Rhinow S., Suendermann D. 2010. *Proc. of the ICAART 2010, 2nd International Conference on Agents and Artificial Intelligence*, Valencia, Spain.

9. Suendermann A., Liscombe J., Dayanidhi K., Pieraccini R. 2009. *Proc. of the SIGDIAL 2009*, London, UK.

10. Ishibuchi H., Nakashima T., Murata T. Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems, in *Trans. on Systems, Man, and Cybernetics,* 1999, vol. 29, p. 601–618.

11. Ward J. Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association*, 1963, № 58, p. 236–244.

УДК 519.8

DISTRIBUTED SELF-CONFIGURING EVOLUTIONARY ALGORITHMS FOR ARTIFICIAL NEURAL NETWORKS DESIGN

D. I. Khritonenko, E. S. Semenkin

Siberian State Aerospace University named after academician M. F. Reshetnev
31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660014, Russian Federation
E-mail: hdmitry.91@mail.ru, eugenesemenkin@yandex.ru

In this paper we describe the method of automatic neural network design based on the modified evolutionary algorithms. The main features of the modification proposed are self-configuration and the usage of distributed computing. Implemented algorithms have been tested on the set of classification tasks. The comparison of the genetic algorithm and the genetic programming algorithm's efficiencies is presented.

Keywords: genetic algorithm, genetic programming, self-configuration, distributed computing, artificial neural network, classifiers, automated design.

РАСПРЕДЕЛЕННЫЙ САМОНАСТРАИВАЮЩИЙСЯ ЭВОЛЮЦИОННЫЙ АЛГОРИТМ ДЛЯ ПРОЕКТИРОВАНИЯ ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ

Д. И. Хритоненко, Е. С. Семенкин

Сибирский государственный аэрокосмический университет имени академика М. Ф. Решетнева
Российская Федерация, 660014, Красноярск, просп. им. газ. «Красноярский рабочий», 31
E-mail: hdmitry.91@mail.ru, eugenesemenkin@yandex.ru

Описывается метод автоматического проектирования искусственных нейронных сетей на основе модифицированного эволюционного алгоритма. Главное свойство модификации заключается в самоконфигурировании алгоритма и распределенных вычислениях. Реализованные алгоритмы были протестированы на множестве задач классификации. Представлено сравнение эффективности генетического алгоритма и генетического программирования.

Ключевые слова: генетический алгоритм, генетическое программирование, самоконфигурирование, распределенные вычисления, искусственные нейронные сети, классификаторы, автоматизированное проектирование.

Self-configuring genetic algorithm. Genetic algorithm (GA) is an optimization method based on genetics and natural selection concepts [1; 2]. The use of GA in a problem solving is accompanied by the problem of relevant algorithm configuration selection.

In every task solving process there is certain optimal configuration (in terms of time consumption, number of calculations, number of fitness function calculation and etc.), which might be altered in process of algorithm functioning. Therefore, our task is to find such configuration to improve the program efficiency.

Flowchart of the self-configuring GA is presented in fig. 1. Presented scheme's difference from standart one [3] is self-configuration section. It allows perform an adjustment of a given configuration in the process of the algorithm execution.

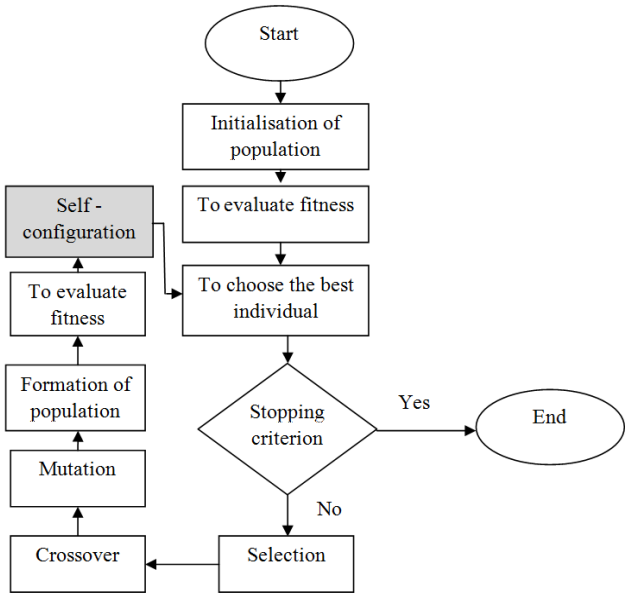

Fig. 1. Flowchart of self-configuring GA

Let us describe proposed adjustment approach. There are 3 groups of GA operators: selection, crossover, mutation. Each group represents a set of the following operators (table 1).

Table 1

Groups of the genetic operators

Selection	Crossover	Mutation
• Proportionate • Ranking • Tournament	• One-point • Two-point • Uniform	• Weak • Average • Strong

Operators shall compete for application in following algorithm iterations.

Let's call as progressive chromosome the chromosome which fitness value surpasses current population's average [4]. It's necessary to make following statement: the number of the progressive chromosome resulted by the use of selection operator is equal to the sum of progressive chromosome resulted from crossover and mutation processes. Probability of operator being chosen is calculated as in following formula:

$$P_i = \frac{q_i}{\sum\limits_j q_j}, \quad (1)$$

where q_i is the number of progressive chromosomes, resulted from i-th operator. On the each algorithm iteration only one operator from each group is used. Therefore, it is crucial to accumulate certain number of iterations to enable further adjustment of operator selection probabilities.

Proposed approach allows adjusting several algorithms' parameters: type of selection/crossover/mutation. However, selection problem of the numerical parameters (such as tournament size) still exists.

Given algorithm has been realized in form of computer program and tested on optimization tasks. Main efficiency criteria of EA is reliability, as it evaluates solution finding probability of given task. The algorithm reliability is a percentage of successful runs when the problem solution is found with required precision.

Since GA is a stochastic algorithm, it's essential to evaluate efficiency by means of several runs. In our case, 100 unaltered program runs have been made for all algorithms. In figure 2, the reliability of conventional GAs with 90 different settings are compared with self-configuring GA on one test problem.

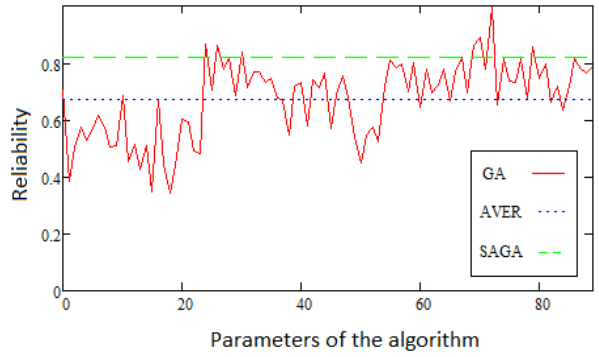

Fig. 2. GA reliability

The figure shows that the modification proposed (self-adjustment – SAGA) surpasses average GAs reliability value (AVER) and not much inferior to the reliability of GAs with the best parameters which resulted from the exhaustive search of settings.

Genetic algorithm in artificial neural network design. In this paper the following encoding method of artificial neural network (ANN) design in GA is presented. Certain number of hidden layers and the maximal number of neurons are given. Number of neurons on each of the layers is encoded in chromosome and adjusted by the methods of GA. Moreover, in each neuron's chromosome specified number (type) of activation function is encoded. Resulting neural network is a perceptron with various activation functions.

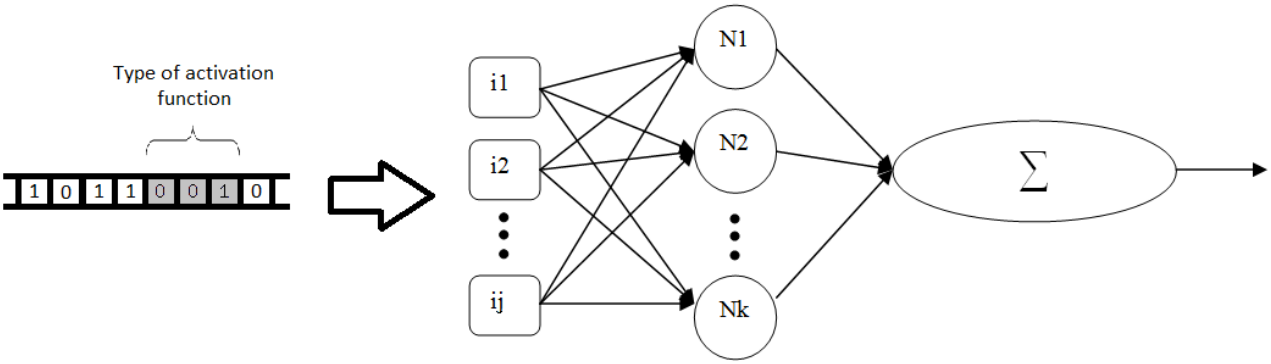

Fig. 3. An example of ANN

In this figure an example of simple ANN encoding by means of binary string is given. In this figure, Nk stands for *k*-th neuron (which is selected from the list of activation function) and ij – j-th input data variable.Presented algorithm has been realized in the form program system (ANN_GA).It's effectiveness and comparison with existing algorithms is presented below. Learning procedure is based on the algorithm presented above.

Self-configuring genetic programming algorithm. Generally, functioning of genetic programming algorithm (GPA) is close to the one of genetic algorithm [5]. The most striking difference is in the encoding procedure. Whilst in GA individual is presented as a binary string of a pre-determined length, GPA uses trees.

The proposed above configuration selection schema has been applied to GPA. Moreover, a modification of the uniform crossover operator [6] is presented in the algorithm. When specified node transportation probability is given only one offspring is created, node sub-trees with different number of arcs compete for usage in crossover. Any another aspect of this algorithm is executed as specified in [7]. An example of this crossover is represented in the figure below.

Presented approach enables to control the trees' complexity. Designed self-configuring GA has been used for the purposes of adjustment of trees' parameters.

Realized as a program system, algorithm has been tested on classification tasks [8]. Australian credit is a task of suspicious bank transactions discovery. This task has 14 variables and 2 classes. German credit is a task of trustworthiness evaluation. This task has 24 variables and 2 classes. Efficiency comparison between proposed algorithm and existing algorithms are shown in the following spreadsheet [9] (table 2).

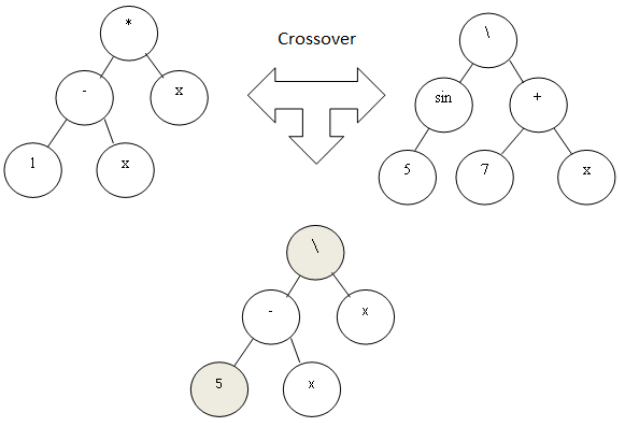

Fig. 4. An example of uniform crossover

Algorithm efficiency criteria:

$$I = 1 - \frac{ERROR}{n}. \quad (2)$$

In this formula ERROR is the number of incorrectly classified objects, n is an overall amount of sample. In tab. 2 average values of 50 runs are given. Sample has been separated into 2 groups (test and training samples). Test sample ratio is 20:80. As represented in previous spreadsheet, proposed self-configuring GPA has shown

moderate results. So, developed self-configuring algorithm in classification tasks has shown results comparable to existing analogues.

Genetic programming algorithm for artificial neural network design. For the purposes of the automated neural network design it's necessary to choose the tree neural network encoding method. Binary trees shall encode neural network. In this algorithm the terminal set embraces all input data variable and neurons with different types of activation function. Functional set shall contain following operators:

"+" – operator which allows to form a layer which includes several neurons;

">" – operator that indicates that "left" sub-tree output data serves as an input data for the "right" sub-tree;

"<" – vice versa.

Illustration of encoding methods is shown below:

In this figure an example of simple ANN encoding by means of tree is given. In this figure, N1 stands for 1-st type neuron (which is selected from the list of activation functions) and ij – j-th input data variable.

Presented algorithm has been implemented in the form of program system (ANN_GP). It's effectiveness and comparison with existing algorithms is presented below. As stated before, GP designed ANN structure learning procedure is based on GA.

Presented algorithms have been tested on classification tasks [8] in conditions specified above. Efficiency criteria are calculated as in the formula (2).

Algorithms have proved high efficiency in solving classification tasks. Worth noting that, ANN_GA and ANN_GP have shown equal efficiency in Wilcoxon signed-rank test. For such purposes 5 series of tests by 10 runs have been conducted. However, GA computing time has been 3 times longer by average. It can be explained by GA encoding multi-layer perceptron with many connections between neurons. Network training in such case is very consuming in terms of resources.

Table 2

The test results

Algorithm name	Australian credit	German Credit	Algorithm name	Australian credit	German credit
SCGP	0,9022	0,7950	Boosting	0,7600	0,7000
MGP	0,8985	0,7875	Bagging	0,8470	0,6840
2SGP	0,9027	0,8015	RSM	0,8520	0,6770
GP	0,8889	0,7834	CCEL	0,8660	0,7460
Fuzzy classifier	0,8910	0,7940	CART	0,8744	0,7565
C4.5	0,8986	0,7773	MLP	0,8986	0,7618
LR	0,8696	0,7837	**GP**	**0,8874**	**0,7697**
k-NN	0,7150	0,7151			

Table 3

The test results

Algorithm name	Australian Credit	German credit	Algorithm name	Australian credit	German credit
SCGP	0,9022	0,7950	Boosting	0,7600	0,7000
MGP	0,8985	0,7875	Bagging	0,8470	0,6840
2SGP	0,9027	0,8015	RSM	0,8520	0,6770
GP	0,8889	0,7834	CCEL	0,8660	0,7460
Fuzzy classifier	0,8910	0,7940	CART	0,8744	0,7565
C4.5	0,8986	0,7773	MLP	0,8986	0,7618
LR	0,8696	0,7837	**ANN_GA**	**0,8994**	**0,7589**
k-NN	0,7150	0,7151	**ANN_GP**	**0,9001**	**0,7596**

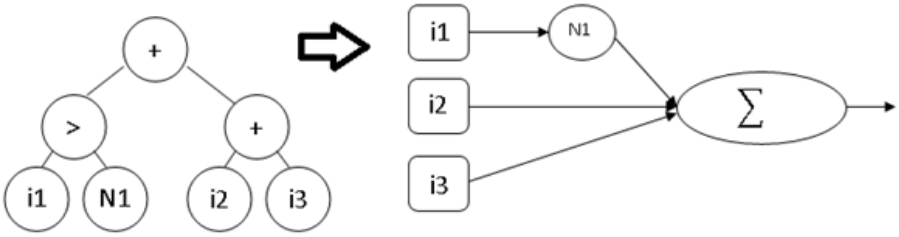

Fig. 5. Neural network encoding example

Distributed computing. EA realization by means of distributed computing application is suggested. It means that all node calculations shall be executed by logic cores of computing systems, while algorithm itself shall not be altered. The main idea is that each one of the threads processes certain groups of individuals.

Worth noting that EA are stochastic. Distribute computing shortens the time-consumption without affecting any other criteria. Such effect has been achieved by means of master-thread usage, which was responsible for random number generator. Wilcoxon signed-rank test as well proves that reliability has not been affected.

Test has been conducted on a PC with following specifications:

– Operating System: Microsoft Windows 7;
– CPU: AMD FX8320 (@3.5GHz);
– RAM: 16GB.

Following graph shows dependency of th time-consumption from a number of logic cores used.

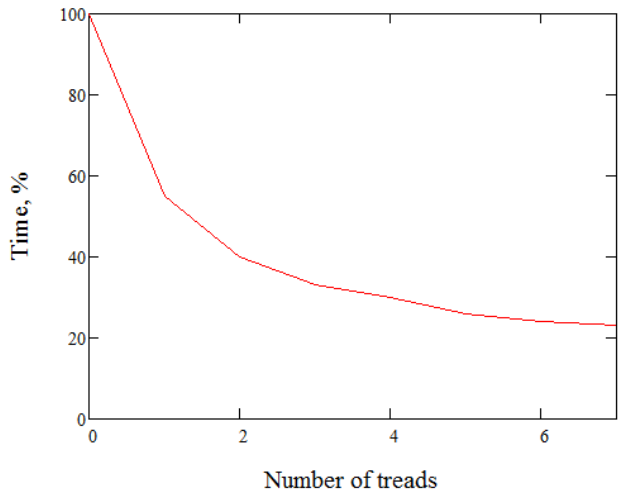

Fig. 6. The test results

Conclusions. Following algorithms have been developed and implemented: distributed self-configuring GA and GPA. Algorithms have been tested on a series of tasks and proved their effectiveness. Additionally, on basis of these algorithms the new artificial neural network design systems have been developed and tested on classification tasks.

Self-configuration enables to solve one of the actual tasks – genetic operator selection in EA. Distributed computing shortens the time-consumption without affecting efficiency. It might be useful in the case that involves great amount of time being spent on calculation. Judging from presented approaches comparison one should conclude that GPA is preferable in cases of artificial neural network design.

References

1. Holland J. H. Adaptation in Natural and Artificial Systems, Ann Arbor: University of Michigan Press, 1975.

2. Holland J. H. Adaptation in natural and artificial systems. MIT Press. Cambridge, VA, 1992.

3. Rutkovskaya D., Pilinsky M., Rutkovsky L. Neural networks, genetic algorithms and fuzzy systems: translation from polish. I. D. Rudinsky. Moscow, Hotline. Telecom, 2006, 383 p.

4. Myasnikov A. S. Insular genetic algorithm with genetic operators probabilities dynamic choice. Available at: http://technomag.edu.ru/doc/136503.html.

5. Koza J.. Genetic Programming. On the Programming of Computers by Means of Nature Selection, 1992.

6. Semenkin E., Semenkina M. Self-Configuring Genetic Programming Algorithm with Modified Uniform Crossover. In: Proc. of IEEE Congress on Evolutionary Computation. IEEE World Congress on Computational Intelligence, Brisbane, Australia, 2012.

7. Poli R., Langdon W. B. On the Ability to Search the Space of Programs of Standard, One-Point and Uniform Crossover in Genetic Programming. Technical Report CSRP-98-7. The University of Birmingham (UK), 1998, 21 p.

8. Machine Learning Repository UCI. Available at: http://archive.ics.uci.edu/ml/datasets.html.

УДК 519.6

SELF-CONFIGURING GENETIC PROGRAMMING ALGORITHM FOR MEDICAL DIAGNOSTIC PROBLEMS

E. A. Popov, M. E. Semenkina

Siberian State Aerospace University named after academician M. F. Reshetnev
31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660014, Russian Federation
E-mail: epopov@bmail.ru, semenkina88@mail.ru

Genetic programming algorithm for neural network automatic design is suggested. Ensemble member competence estimations based procedure of decision making with intelligent information technologies is proposed. Effectiveness of the approach is proved on the benchmark and real-world medical diagnostic problems.

Keywords: intelligent information technologies, genetic programming algorithm, ensemble based decision making.