Proceedings of the 6th Győr Symposium and 3rd Hungarian-Polish
and 1st Hungarian-Romanian Joint Conference on Computational Intelligence

Tuning Neural Networks with Krill Herd Algorithm

P.A. Kowalski[1,2], **S. Łukasik**[1,2]

[1]**Systems Research Institute, Polish Academy of Sciences**
ul. Newelska 6, PL-01-447 Warsaw, Poland,
E-mail: pakowal, slukasik@ibspan.waw.pl

[2]**Department of Automatic Control and Information Technology,**
Cracow University of Technology
ul. Warszawska 24, PL-31-155 Cracow, Poland

Abstract: In recent times, several new metaheuristic algorithms based on natural phenomena have been made available to researchers. One of these is that of the Krill Herd Algorithm (KHA) procedure. It contains many interesting mechanisms. The purpose of this article is to compare the KHA optimization algorithm used for learning an artificial neural network (ANN), with other heuristic methods and with more conventional procedures. The proposed learning method has been verified positively for the classification problem. For this task, benchmark examples drawn from the Irvine Machine Learning Repository were used. For comparison purposes, both the Classification Error and Sum of Square Errors were employed.

Keywords: *Krill Herd Algorithm, Biologically Inspired Algorithm, Metaheuristic, Neural Networks, Optimization.*

1. Introduction

Increasingly, in the engineering profession, optimization methods and algorithms are becoming essential tools. However, employing these is time-consuming. This is due to the need for extensive computational power when deriving solutions through their enlistment, and rests as well in the nature of the properties of the employed methods and algorithms, themselves. The methods that are currently used (to very good effect) in deriving solutions to problems of optimization, are the gradient methods and the heuristic algorithms. Unfortunately, both procedures, in addition to their advantages, also have some drawbacks.

The advantages of employing gradient methods lies in their ability to enable the achieving of a rapid convergence to the nearest optimum, the result of which may not always be the global optimum. Moreover, when used for deriving a solution to a multi-modal function, by way of their enlistment, very often a local optimum is found. The disadvantages of employing these methods rests within the conditions of the objective function. This must be continuous. What is more, the Hessian function must be positive definite, while the calculations are performed with a single starting point, which, in turn, significantly restricts a search area. Furthermore, the choice of the starting point has an impact on the convergence of the method, and may bring about the possibility of the results falling into a local extreme. Many local-minimization procedures e.g. the Back-propagation Algorithm or the Quasi-Newton methods for optimization tasks, can be applied (in particular, for the learning of neural networks) [14].

Metaheuristic procedures for finding the global optimum (usually with a certain probability) are called global optimization methods. To this group belong the Evolutionary Algorithms [6], the Simulated Annealing Algorithms [9], the Immunological Methods [2] and the swarm intelligence procedures [10]. The aforementioned heuristic algorithms require only knowing the formula of the cost (fitness) function, and are quite simple to implement by way of today's computers. Nowadays, the group of algorithms that belong within the category of swarm intelligence is very extensive. Among the latest Nature-inspired metaheuristics are the Harmony Search [4], the Firefly Algorithm [8], the Cuckoo Search [15], the Flower Pollination Algorithm [11] and the Krill Herd Algorithm [3].

Heuristic methods are often used for neural network learning process, and they are an alternative methods to such traditional gradient algorithms as, for instance, error Back-Propagation or Levenberg-Marquardt procedures. However, for certain types of a neural network, these methods cannot be used. This comes about due to the lack of the possibility of applying analytical derivative formulas, or because a small convergence is achieved within the learning algorithm. The Fuzzy Flip-Flop neural network type is an example of both aforementioned phenomena. For this type of learning, certain neural network algorithms have been applied, among them, the Bacterial Memetic Algorithm [7] and the Evolutionary Strategy [6]. For a typical network of a multilayer perception type, many heuristic optimization methods can be also employed. These range from the genetic algorithm, to the evolutionary algorithms and end with the swarms procedures. From the results found within scientific publications, it can be concluded that very often after using the heuristic algorithm, we obtain positive results much faster than when applying gradient methods.

The use of the Krill Herd Algorithm have become very popular recently. This is because it is an effective modern optimisation and search procedure [3]. This metaheuristic technique is based on the behaviour of a krill herd. The purpose of this paper is to

investigate the possibility of applying the Krill Herd Algorithm for parameters optimization purposes within artificial neural networks (ANN). In our work, the results of numerical studies based on the classic examples of benchmark data, were compared with other heuristic methods, and with a gradient algorithm.

The outline of this paper is as follows. After a short Introduction, in Section 2, information about the Krill Herd Algorithm (KHA) is to be introduced, while in Section 3, that of the neural network and the tuning of its structure and parameters by way of using the KHA will be presented. Subsequently, In section 4, application examples will be presented. Finally, a conclusion will be drawn in last part of this work.

2. Krill Herd Algorithm

In this chapter, the optimization algorithm covered by this paper will be briefly described. KHA is one of the newest optimization procedures that come with a heuristic character. Its main inspiration lays in following and imitating the biological swarming behaviour of the Antarctic Krill (*Euphausia superba*), found in the Southern Ocean. This algorithm was introduced, in 2012, by A. H. Gandomi and A. H. Alavi [3].

The Krill metaheuristic is used in solving optimization tasks. This consists of finding the extreme point for the function f, called the 'fitness function' or the 'cost function'. In essence KHA procedure is based on observing such behaviours in the herd as foraging and communicating with other members of the swarm. Therefore, the position of the particular individuals $(i = 1,\ldots,M)$ in the herd is described through the following equation:

$$dX_i/dt = N_i + F_i + D_i, \tag{1}$$

The Lagrangian model (1) contains three components: N_i denotes the motion induced by other individuals within the swarm, F_i is the foraging motion and D_i provides the physical diffusion of i-th krill. The movement effected by the presence of other krill can be described using the following formula:

$$N_i = N^{max}\alpha_i + \omega N_i^{old}. \tag{2}$$

In equation (3), individuals try to maintain a high density within the herd and move in a direction induced by the α_i parameter. This, is calculated taking into account local effects such as swarm density, and also a global character that is based on the best krill position. Here N^{max} denotes the maximum induced speed. Additionally, parameter ω, the induced motion inertia weight, is present. This quantity reveals the importance of the N_i^{old} values

from the previous iteration, when determining the new value N_i. Information on the values of the particular parameters listed here can be found in [3].

The foraging motion includes two main components:

$$F_i = V_f \beta_i + \omega_f F_i^{old}, \tag{3}$$

where the first one is one in which an individual krill informs the other swarm members about a new food source location, while the other one describes the swarm's previous experience with respect to prior food location. Furthermore, quantity V_f describes the speed of searching for food, and has been selected empirically. Its recommended value is 0.02 [3]. The location of a food source is the quantity that for KHA, is dependent upon the basis of the distribution of the fitness function and the individual krill position. It is given by following equation:

$$X^{food} = \frac{\sum_{i=1}^{M} \frac{1}{f_i} X_i}{\sum_{i=1}^{M} \frac{1}{f_i}}. \tag{4}$$

The physical diffusion part for each individual is an introduced random factor which is formulated as follows:

$$D_i = D^{max} \left(1 - \frac{I}{I_{max}}\right) \delta \tag{5}$$

Here, I is number of the number of current iteration, I_{max} indicates a maximum number of iterations, $D^{max} \in [0.002; 0.1]$ represents the maximal diffusion speed, while notation δ represents the random directional vector, with its elements belonging within the interval $[-1; 1]$.

Finally, for each krill, its location at time $t + \Delta t$ is determined as follows:

$$X_i(t + \Delta t) = X_i(t) + \Delta t \frac{dX_i}{dt} \tag{6}$$

wherein Δt is the scaling factor for the speed of the search of the solution space.

At the last stage of the KHA procedure, genetic operators are used [13]. Doing so, primarily classical mutation and crossover operators known from Genetic Algorithms are employed. In other studies [12], alternative operators based on Differential Evolution are proposed. This phase is optional, implementation of these operators can be completely

omitted or only one of them can be employed. As shown by certain preliminary tests [3], the Krill Herd Algorithm achieves the best results with an implemented differential crossover operator. Generally, the KHA procedure can be described by an introduced flowchart as in Figure 1. Additional information about the Krill Herd Algorithm can be found in [5, 13].

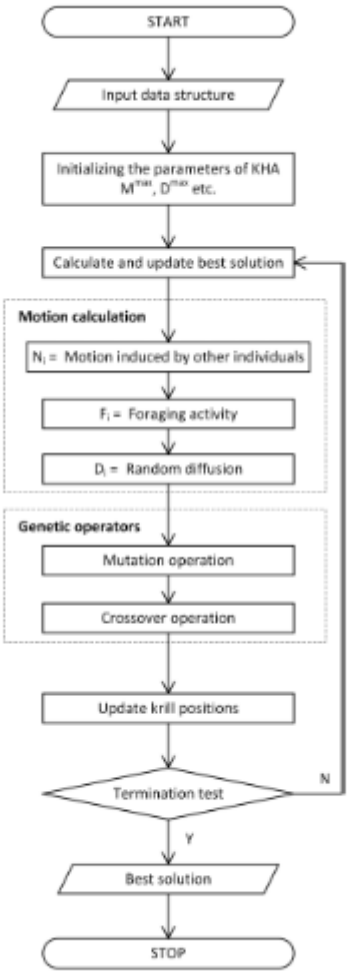

Figure 1. Flowchart of KHA.

3. KHA's Application for Tuning a Neural Network's Parameters

An optimization procedure based on the KHA described in section 2 has been applied for the training of an Artificial Neural Network (ANN), and the obtained results were compared with the ones obtained by using: Back Propagation (BP), the Genetic Algorithm (GA) and the Harmony Search (HS) taken from article [4]. In the KHA training method, all weights and biases from ANN translate in to a vector. This represents the position of an individual krill. In our study, the initial conditions for a krill position are obtained using a random value generator with uniform distribution. Moreover, in the presented research, the parameters of KHA were adopted based on literature [3] and [5]. In particular, the following parameters were assumed: $N^{max} = 0.01$, $V_f = 0.02$, $\omega = 2.0$ and finally $D^{max} = 0.7$. In addition, for each example, the scaling factor associated with element dt was determined individually. Here the training process is terminated when maximal epoch number has been reached. In this study, it was set as 100.

For the purposes of ascertaining the fitness of the studied algorithms, as the value of the fitness function, a Sum of Square Error over training (SSE) was considered. However, for the classification task, a more representative practice is using the Classification Error (CE), hence, naturally it was selected as a second quality measure:

$$CE = \frac{err_T}{P} 100 \tag{7}$$

were err_T denotes the total number of misclassifications and P is the number of examples under classification. It should be clearly emphasized that the choice of both measures is not accidental. This because generating the minimum value of SSE error does not always entail a marked decrease in the CE error. This can be explained by considering the construction of a neural network. Here, the output layer is a set of neurons representing each of the considered classes. The result of classification is, hence, designated by way of an index of neurons of the highest output value, thus reducing - during the learning phase - the value of the output signals, which in turn decrees the overall SSE error, while not always having an impact on the CE error. To circumvent this, sometimes a weighted error containing both types of errors (SSE and CE) is employed. A thread associated with the selection of an appropriate measure of the classification error, will be the subject of other extended research and publications.

4. Numerical Experiments

For the numerical verification of the proposed algorithm, a classification of selected benchmark data was carried out. These data sets come from the well-known UCI Machine

Learning Repository [1]. Detailed information related to the cardinality of the learning and testing sample, the number of classes, as well as the dimensionality and the structure of the artificial neural network, can be found in Table 1.

Table 1. Data sets used for verification

Data set	No. Attributes	No. Classes	Training examples	Testing examples	Equinumerosity examples in classes	No. neurons in hidden layer	No. NN Parameters
IRIS	4	3	120	30	Yes	5	43
IONOSPHERE	33	2	281	70	No	4	146
GLASS	9	6	171	43	No	12	198
THYROID	21	3	5760	1440	No	15	378

Additionally, it should be emphasized here that the last two sets are characterized by having a very large imbalance in the multiplicity of examples within the considered classes. For the classification of the individual data sets, ANN were constructed. Their structure is based on the comparative examples presented in article [4]. Thus, all tested networks have three layers, and all neurons have an applied $tanh$ transfer function. Moreover, input-output data values were normalized to be in the range $[-1,1]$.

In order to generate a comparison with the other results, the data sets were divided up as follows: 80% of all samples constituted learning data, while the rest represented the testing data. Naturally, these proportions are maintained for each of the classes. For every data set, the main learning algorithm was executed 20 times, using the KHA procedure with the same parameters as described in the previous section. The exception here is a factor for a scaling step associated with δt. This was individually set for each data type, and was used as 2.0,3.0,3.7,3.0 respectively.

All parameters used in this investigation were based on recommendations that were found in the literature of [5] and [3]. What is more, the selection variable factor was based on the pilot runs tests. In the case of the first two sets of benchmarks (ie. Iris and Ionosphere), the cardinality of the population was 50, while in other cases, this figure was set at 70 individuals.

In table 2 all numerical results are included. Furthermore, the outcomes obtained were divided into two parts. The first of these is related to the neural network learning process, and therefore it contains information about the SSE errors and the CE errors, in addition to the number of iterations (epochs) in which these errors were archived. The second part of the table, consists of the test results received from the best networks.

Table 2. Results of the learning and testing process

Data set	Method	Learning			Testing	
		SSE	CE	Accepted Iteration	SSE	CE
IRIS	KHA	21. 28	**0.41 %**	30	4.88	**0.00%**
	HS	18.00	1.67 %	162	—	3.33 %
	GA	96.00	10.00 %	66	—	10.0 %
	BP	7.85	0.83 %	1254	—	3.33 %
IONOSPHERE	KHA	**31.0**	11.00 %	**85**	13.87	8.57 %
	HS	106.4	5.00 %	170	—	5.63 %
	GA	152	6.79 %	2244	—	5.63 %
	BP	8.52	0.56 %	1628	—	4.23 %
GLASS	KHA	41.21	40.94 %	21	9.82	41.86 %
	HS	355.85	29.82 %	177	—	27.91 %
	GA	544.00	42.11 %	6123	—	32.56 %
	BP	218.06	18.71 %	662	—	32.56 %
THYROID	KHA	**320.3**	**5.19 %**	75	35.9	**7.10 %**
	HS	3146.4	6.94 %	94	—	7.22 %
	GA	3416.0	7.42 %	167	—	7.43 %
	BP	450.0	1.33 %	4201	—	2.78%

5. Discussion of Results and Conclusions

In the case of the well-known Iris Set, a neural network which unmistakably classifies the elements of the test set was obtained. At the same time, worth mentioning is the presence of the smallest CE error within the table of generated results. This lies within the learning phase, and it amounts to 0.83%. Moreover, it constitutes one element of the 120 examples contained within this part of the data. This result is compared with the BP method, but has been obtained after 30 iterations, and not, as in the case of a competitive algorithm, in 1254.

The original Ionosphere Data contains 34 elements in the feature vector, but one coordinate in all cases was found to be equal to 9, therefore, it was removed from the data set. Accordingly, a neural network with a smaller structure $33 - 4 - 2$ was examined. In this particular case, the first class is represented by 64% of the examples, so it can be seen that this data set is unbalanced. In the learning process, the lowest SSE error type (within heuristic method groups) was achieved. This, amounted to 31.0. On the other hand, CE and testing errors are not so impressive. However, the complexity of the data may indicate the time of learning that comes about through using the other methods.

The Glass Data set represent 6 types of glass. In this case, it must be emphasized that the first and second class contain 70% of all the data's examples. In dealing with this set, the proposed algorithm obtained the smallest SSE error of learning, and, in addition, the

training process lasted only 21 epochs. Moreover, the set's CE error is smaller than in the case of employing AG, but greater than that of the other methods.

The last of the considered data sets was called Thyroid. In this case, we have a very complicated data set because the first class consists of 92.57% examples. What is more, the neural network has been designed with a $21 - 15 - 3$ structure. This results in that the optimized problem is represented by 378 parameters. In these tests, we obtain the best results for the learning process, and with respect to the results of KHA testing, we achieved the best result of heuristic methods that were examined.

This algorithm, although it may seem quite complicated, can be used to very quickly obtain satisfactory results. Hence, we believe that the proposed method is often the best one to be employed, and is almost always the best of the heuristic algorithms that can be used in classifying data sets. However, in some cases, the classical gradient method (BP) was shown to yield better results, but the time of its execution was incomparably longer. Moreover, during the numerical study, we discovered that the KHA procedure showed a fairly sizeable sensitivity to some internal parameters. This subject was initially considered in [5]. In addition, the selection of formula of fitness function for individual evaluation is very important and sensitive so as to obtain best results. This last will be the subject of a separate article.

Acknowledgement

This research was supported in part by PL-Grid Infrastructure.

References

[1] Asuncion, A., Newman, D.J.: *UCI Machine Learning Repository* [http://www.ics.uci.edu/ mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science, 2007.

[2] Castro, L. N., Timmis, J.: *Artificial Immune Systems: A New Computational Intelligence Approach*, Springer, 2002

[3] Gandomi, A.H., Alavi, A.H.: *Krill herd: A new bio-inspired optimization algorithm*, Communications in Nonlinear Science and Numerical Simulation, vol. 17, pp. 4831–4845, 2012

[4] Kattan, A., Abdullah, R.: *Training Feed-Forward Artificial Neural Networks For Pattern-Classification Using The Harmony Search Algorithm*, in The Second Interna-

tional Conference on Digital Enterprise and Information Systems (DEIS2013), The Society of Digital Information and Wireless Communication, pp. 84-–97, 2013

[5] Kowalski P.A., Łukasik S.: *Experimental Study for Selected Parameters of the Krill Herd Algorithm*, in Advances in Intelligent Systems and Computing, 2014 (in press)

[6] Kowalski, P.A.: *Evolutionary Strategy for the Fuzzy Flip-Flop Neural Networks Supervised Learning Procedure*, in Articial Intelligence and Soft Computing, Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (Eds.), Lecture Notes in Computer Science vol 7894, Springer Berlin Heidelberg, pp. 294–305, 2013

[7] Lovassy, R., Koczy, L.T., Gal, L.: *Optimizing Fuzzy Flip-Flop Based Neural Networks by Bacterial Memetic Algorithm* in IFSA/EUSFLAT, Lisbon, pp. 1508-–1513, 2009

[8] Łukasik, S., Zak, S.: *Firefly Algorithm for Continuous Constrained Optimization Tasks*, in Computational Collective Intelligence, Semantic Web, Social Networks and Multiagent Systems, Nguyen, N.T., Kowalczyk, R., Chen, S.M. (Eds.), Lecture Notes in Computer Science vol. 5796, Springer Berlin Heidelberg, pp. 97–106, 2009

[9] Łukasik, S., Kulczycki, P.: *An Algorithm for Sample and Data Dimensionality Reduction Using Fast Simulated Annealing*, in Tang, J., King, I., Chen, L., Wang, J. (eds.) ADMA 2011, Part I. LNCS (LNAI), vol. 7120, pp. 152—161. Springer, Heidelberg, 2011

[10] Łukasik, S., Kowalski P.A.: *Fully Informed Swarm Optimization Algorithms: Basic Concepts, Variants and Experimental Evaluation*, in 2014 Federated Conference on Computer Science and Information Systems (FedCSIS), 2014 (in print)

[11] Łukasik, S., Kowalski P.A.: *Study of Flower Pollination Algorithm for Continuous Optimization*, in: Advances in Intelligent Systems and Computing, 2014 (in press)

[12] Wang, G.G., Gandomi, A.H., Alavi, A.H., Hao, G.S.: *Hybrid krill herd algorithm with differential evolution for global numerical optimization*, Neural Comput & Applic pp.1-–12, 2013

[13] Wang, G.G., Guo, L., Wang, H., Duan, H., Liu, L., Li, J.: *Incorporating mutation scheme into krill herd algorithm for global numerical optimization*, Neural Comput & Applic, vol.24, pp. 853-–871, 2014

[14] Werbos, P.J.: *The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting*, Wiley-Interscience, New York, 1994

[15] Yang, X.S., Deb, S.: *Cuckoo search: recent advances and applications*, Neural Computing and Application, pp. 1—6, 2013