# Neural Networks Training by Genetic Algorithm with Adjusted Crossover on Pattern Classification

**Mário Tasso Ribeiro Serra Neto**
Centro Universitário do Estado do Pará, Belém, Pará, Brasil. Grupo de Estudos Temático em Computação Aplicada - Laboratório de Inteligência Computacional (LINCOMP - CESUPA)
mariotrsn@gmail.com

**Demison Rolins de Souza Alves**
Centro Universitário do Estado do Pará, Belém, Pará, Brasil. Grupo de Estudos Temático em Computação Aplicada - Laboratório de Inteligência Computacional (LINCOMP - CESUPA)
demisonalves9@gmail.com

**Fabio dos Santos Ferreira**
Centro Universitário do Estado do Pará, Belém, Pará, Brasil. Grupo de Estudos Temático em Computação Aplicada - Laboratório de Inteligência Computacional (LINCOMP - CESUPA)
ferreira06@gmail.com

**Edson Koiti Yasojima**
Federal Rural University of Amazônia (UFRA)
Electrical Engineering Post Graduation Program of the Federal University of Pará (UFPA)
koitiyasojima@gmail.com

**Rodrigo Lisbôa Pereira**
Federal Rural University of Amazônia (UFRA)
Electrical Engineering Post Graduation Program of the Federal University of Pará (UFPA)
rod.lisboa@ieee.org.br

**ABSTRACT**

Countless analysis of Evolutionary Algorithms applied for training of neural networks suggest that they are superior in both processing time and robustness than exact methods for small sized networks. Genetic Algorithms are from the family of Evolutionary Algorithms, well-known for being efficient for solving many classes of problems. In this paper, we propose an adaptation of an efficient crossover method used for constrained real valued optimization called KX to train neural networks. The operator considers the direction of which each variable of a candidate solution tends to improve to guide the offspring towards a possible optimum, boosting the convergence towards optima. The modified GA is tested on four different datasets and compared against other population and gradient based methods. Results indicate that this crossover is best suitable for small networks with few outputs, a family of problems where Evolutionary Algorithms dominate over gradient based methods.

**KEYWORDS. GENETIC ALGORITHM, CROSSOVER, NEURAL NETWORKS.**

**Paper topics: MH – Metaheuristics.**

https://proceedings.science/p/85564?lang=pt-br

## 1. Introduction

Neural networks for Pattern classification or Deep Learning problems often features dozens or hundreds of thousands of connections weights. Because of that, gradient based methods overshadow Evolutionary Algorithms (EA) due to their capabilities of better handling the high dimensionality [Karaboga and Ozturk, 2009; Yasojima et al., 2016]. However, for problems with less than 1000 connection weights, EA's are significantly more robust than any exact method [Morse and Kenneth, 2016].

The Genetic Algorithm (GA) is an EA that is standard at solving combinatorial and real valued optimization problems. With the objective of improving the performance of the GA for training neural networks, and at the same time trying to achieve better efficiency at higher dimensionality problems, an adaptation of the crossover operation of the GA, called KX crossover is introduced. The KX crossover considers the direction of which each variable of a candidate solution tends to improve to guide the offspring towards a possible optimum, boosting the convergence towards optima.

Performance of the approach is verified on four representative benchmarks with both single and multiple inputs and outputs and dimensions ranging from hundreds to thousands. The Genetic Algorithm with the KX crossover is tested against a canonical GA, the Artificial Bee Colony, a prominent algorithm used for training neural networks; an improved version of the ABC designed specifically for training medium sized networks; the canonical Backpropagation (BP) together with an optimized version of the BP. Results assert the well-known statement that EA's are better suited for small-scale networks featuring less than 1000 weights. However, the KX crossover fared better than others population based methods, suggesting that the operator further improved the performance of the GA for instances of the problems that they already have advantages over other techniques.

The paper is organized as it follows; Section 2 gives a background of neural networks, while Section 3 briefly explains what are Genetic Algorithms. Section 4 describes the KX crossover while Section 5 details the adaptation made to the crossover and the algorithm to train neural networks. Section 6 illustrates the settings of the experiment, and the results are discussed in Section 7. Finally, Section 8 delineates conclusions about the experiment and highlights future works.

## 2. Training a Feed Forward Neural Network

On supervised learning, when classifying a dataset using Artificial Neural Networks (ANN), data are propagated along the architecture to obtain the best collections of weights that best match its architecture. After the work of Rumelhardt, Williams and Hinton [Rumelhart et al., 1986] in the 1980s, backpropagation became the standard method of training ANNs, leading to a thorough study of the field and bringing remarkable improvements with adaptations of the original BP. To perform the training of a network, it is necessary to: initialize an array of weights of size equivalent to the amount of inputs of the neurons; propagate the weights along the nodes of the previously defined architecture; and finally, process the output result [Karaboga et al., 2007; Karaboga and Ozturk, 2009]. The output processing of a neuron is mathematically represented by (1),

$$y_i = f_i\left(\sum_{j=1}^{n} w_{ij}x_j + b_i\right) \tag{1}$$

where xj is the input of the jth neuron, wij is the weight corresponding to the connection between the ith and jth neuron, and bi is the corresponding bias of layer [Rumelhart et al., 1986; Karaboga et al., 2007]. During the training phase of the network, the bias must be treated like a new weight that can be altered or not during the process.

https://proceedings.science/p/85564?lang=pt-br

The output of the last layer of the network transfers the data to an activation function, characterized as nonlinear. Although activation functions can come in many different forms, the most prominent examples can be found in [Hagan et al., 2013].

The performance of the network can be measured by several cost estimator functions. However, the mean squared error (MSE) function, represented by Eq. (2), is usually used:

$$C(w,b) = \frac{1}{2n}\sum_{x} \| \, y(x) - a \, \|^2 \tag{2}$$

where w is the collection of all weights and b is the bias of the network. The function measures the square of the error of the weights and bias found in the ANN. Therefore, the goal is to minimize the MSE by optimizing the weights so that C(w, b) ≈ 0 [Nielsen, 2015].

Aside from Backpropagation, many other techniques are used to train neural networks, such as exact methods and heuristics [Karaboga and Ozturk, 2009]. Heuristic methods usually involve using Evolutionary Algorithms (EA) to train the networks. EAs have been shown to be a good alternative to exact methods for networks whose size is relatively small (less than 1000 connection weights) [Morse and Kenneth, 2016]. The Genetic Algorithm is an example of such that is widely used in combinatorial optimization problems as well as real number optimization problems.

## 3. Genetic Algorithm

Genetic Algorithms are population based optimization algorithms from the family of EA's that solve problems of combinatorial optimization and global optimization with and without restrictions. These problems are often those that require a search through a wide space of candidate solutions [Goldberg, 1989; Eberhart and Simpson, 1996]. GA being a population based algorithm means that it uses samples of a finite pool of candidate solutions, called "population". Each individual of the population is considered as a possible solution to the predefined problem, where each of which has a linear list of alleles and a fitness value. Based on the fitness value and genetic operators, a new population is generated iteratively [Juang, 2004].

Defined by Holland [1975], GA consists of three fundamental stages: evaluation; reproduction and replacement of the population. The second step is divided into: selection, crossover and mutation [Holland, 1975; Goldberg, 1989; Juang, 2004]. Many different approaches were created for each phase depending on the nature of the problems that the GA has to optimize. According to Haupt and Haupt [2000], selection is responsible for selecting individuals from the population, using, for the most part, the value of fitness as criterion. For this stage of the GA, it is also possible to use the roulette or tournament selection method to select individuals.

The mutation operator is considered to be a mechanism for random search throughout the search space that forces candidate solutions out of local optima. This operator introduces small changes in individuals' chromosomes by randomly and with a small probability of changing the content of a gene by a random allele [Eberhart and Simpson, 1996; Juang, 2004].

It is through the crossover operation that the algorithm explores the search surface, within the alleles that are in the individual population, which leads to convergence of results [ Haupt and Haupt, 2000]. This operation consists in the exchange of pieces of the chromosome between the individuals selected as parents, for the creation of new individuals better adapted. Traditional methods such as: one-point crossover and two cut-off points can be implemented [Holland, 1975; Haupt and Haupt, 2000; Juang, 2004].

Due to the importance of the crossover operation and the lack of significant results on real valued constrained minimization problems, Yasojima et al. [2016] devised a crossover technique called KX crossover. This crossover aims at the difference of the values between the alleles of the individuals to provide a direction to which the candidate solutions can converge.

https://proceedings.science/p/85564?lang=pt-br

## 4. KX Crossover

The proposed crossover by Yasojima et al [2016] combines the genes of two individuals and create two new offspring with the adjusted genes of each corresponding parent. After the selection phase of the genetic algorithm, an array based on the genes of all selected individuals ordered by fitness value (ascending) is made. This array indicates the best possible direction of that gene. Figure 1 details the crossover process.

For example, if the first variable value is increasing as far as the individuals are improving their fitness (based on the selected individuals mentioned before) then the direction of that variable is set to 1, meaning that it tends to increase, creating better individuals. If the gene tends to decrease, a value of -1 is assigned for that specific variable.

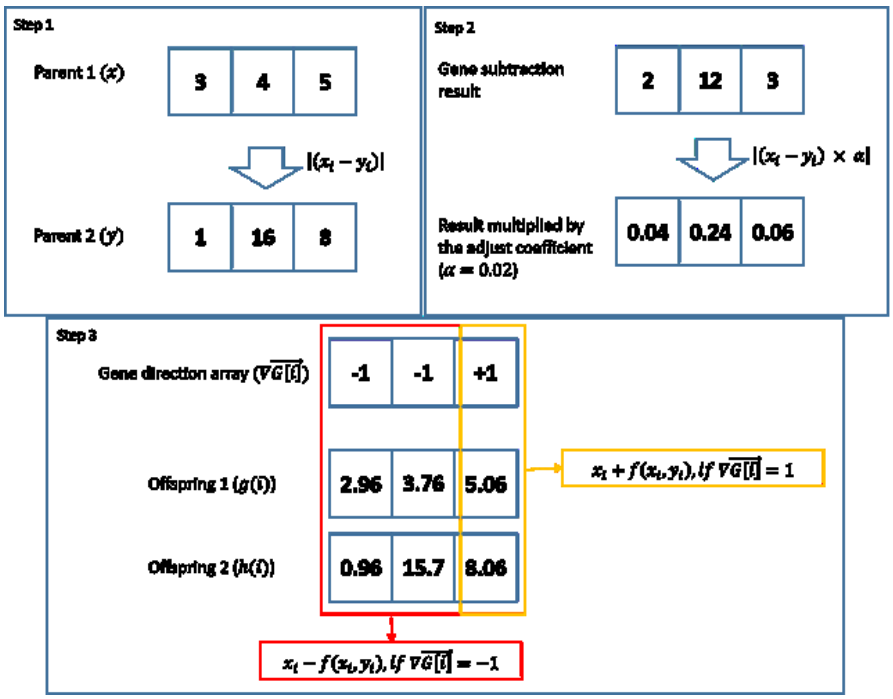



**Figure 1**. Illustration of the KX Crossover Algorithm

So, if a parent with genes P1 = {x1, x2, …, xi}is selected, and another parent with genes P2 = {y1, y2, …, yi}, the two resulting offspring will be X1 = {g(1), g(2), …, g(i)} and X2 = {h(1), h(2), …, h(i)} where (3):

$$f(x_i, y_i) = |(x_i - y_i) \times \alpha| \tag{3}$$

$$g(i) = \begin{cases} x_i + f(x_i, y_i), & if\ \overrightarrow{\nabla G[i]} = 1 \\ x_i - f(x_i, y_i), & if\ \overrightarrow{\nabla G[i]} = -1 \end{cases}$$

$$h(i) = \begin{cases} y_i + f(x_i, y_i), & if\ \overrightarrow{\nabla G[i]} = 1 \\ y_i - f(x_i, y_i), & if\ \overrightarrow{\nabla G[i]} = -1 \end{cases}$$

where α is an adjustment coefficient for the offspring, it is set as a real value ranging between the interval (0,1). Higher values have greater influence on the adjustment, while values close to 0 have almost no influence whatsoever.

https://proceedings.science/p/85564?lang=pt-br

## 5. Approach

Guidelines to train neural networks using population based algorithms are abundant, examples of such can be found in [Prechelt, 1994; Weiß, 1994; Juang, 2004; Karaboga and Ozturk, 2009]. Contrary to the canonical guidelines, the model proposed treats the candidate solutions as a vector with size defined by (4),

$$Size = (QI * NI) + (NI * NO) + NI + NO \quad (4)$$

where QI corresponds to the number of inputs of the problem, NI to the number of neurons in the intermediate layer and NO the number of neurons in the output layer. To evaluate each solution, the weights of the array are sent to the weight matrix that are propagated by the ANN architecture to find the- cost of each solution evaluated by the error function. For this paper, the MSE function was chosen by the authors.

A sigmoidal activation function represented by (5) is used to normalize the outputs of each layer.

$$f(x) = \frac{1}{1 + e^{-x}} \quad (5)$$

The output layer was encoded with the hot-encode technique, used by [Karaboga and Ozturk, 2009; Bullinaria and AlYahya, 2014, Nielsen, 2015], where each distinct class of the problem is controlled by a different output neuron. These are responsible for the result obtained as fitness value. Algorithm 1 demonstrate step by step the KX crossover.

1: Initialize *N* Individuals
2: Evaluate each Individual
3: **while** generation < maxGenerations, **do**:
    3.1: **for** each individual in the population, **do**:
        3.1.1: **if** a random number (0, 1) < Crossover Rate, **do**:
            3.1.1.1: Select 2 random individuals (j, k)
            3.1.1.2: Apply tournament between (j, k) and select the best one (b)
            3.1.1.3: Apply KX Crossover between individuals (i. b)
            3.1.1.4: Evaluate i and b childrens by Propagating
            3.1.1.5: **if** Children (i) fitness < Father (i) fitness, **do**
                3.1.1.5.1: Father (i) fitness, Solution = Children (i) fitness, Solution
            3.1.1.6: **end if**
            3.1.1.7: **if** Children (b) fitness < Father (b) fitness, **do**
                3.1.1.7.1: Father (b) fitness, Solution = Children (b) fitness, Solution
            3.1.1.8: **end if**
        3.1.2: Apply GA Mutation
    3.2: **end for**
    3.3: Search for the global best individual
4: **end while**
5: Calculate the Accuracy through the best individual

**Algorithm 1**. Genetic Algorithm with Adjusted Crossover for Neural Networks

## 6. Experiment

Four datasets, Thyroid, Heart, Soybean and Digits, can be found in Proben1 [Prechelt, 1994] and UCI Machine Learning [Blake, 1998]. These were selected for this experiment because of their distinct topologies when building the network, generating diversity in the amount of

https://proceedings.science/p/85564?lang=pt-br

weights that need to be optimized to better classify a dataset. Each dataset was separated in training and test data as shown in Table 1.

| Dataset | Architeture | Weights | Training | Testing |
|---|---|---|---|---|
| Thyroid | 21-20-3 | 503 | 5400 | 1800 |
| Heart | 35-20-2 | 762 | 690 | 230 |
| Soybean | 82-20-19 | 2059 | 513 | 170 |
| Digits | 64-40-10 | 3010 | 3.823 | 1.797 |

**Table 1.** Configurations of each Dataset

Thyroid is the dataset of diagnosis of thyroids, verifying whether there is hyper or hypo function of each patient. In this dataset, 21 Inputs are used to classify 3 classes of patient's thyroid function as being over function, normal function, or under function. It is the largest dataset found in Proben1 [Prechelt, 1994; Karaboga and Ozturk, 2009].

Heart dataset contains 920 patterns, each pattern has 35 inputs and 2 outputs. The challenge is to classify the diagnosis of a heart condition (if at least one of four major vessels is reduced in diameter by more than 50%). The data set contains 920 patterns, each pattern has 35 inputs and 2 outputs [Prechelt, 1994; Karaboga and Ozturk, 2009].

Soybean data set has the highest number of classes to identify among 19 different soybean diseases. The classification is based on a description of the bean (e.g. whether its size and color are normal) and the plant (e.g. the size of spots on the leafs, whether these spots have a halo, whether plant growth is normal, whether roots are rotted) plus information about the history of the plant's life (e.g. whether changes in crop occurred in the last year or last two years, whether seeds were treated, how the environment temperature is) [Karaboga and Ozturk, 2009; Prechelt, 1994].

According to Bullinaria and AlYahya [2014], some of the datasets used in [Karaboga and Ozturk, 2009] are trivial, because even algorithms with low complexity tend to perform well. To solve this problem, the Digits set was inserted in order to compare the results of the algorithms proposed by Bullinaria and AlYahya [2014] as a more challenging dataset than the problems found in [Karaboga and Ozturk, 2009]. Handwritten Digits dataset that has 64 inputs representing pixelated images and 10 output classes for the digits 0–9, with 3,823 training patterns and 1,797 for testing [Blake, 1998; Bullinaria and AlYahya, 2014].

### 6.1. Settings

Gradient and population-based optimization algorithms were selected to compare the performance of the KX Crossover when inserted into GA. These are: 1) Canonical Genetic algorithm developed by Holland [1975]; 2) Backpropagation presented by Rumelhart et al. [1986]; 3) Artificial Bee Colony proposed by [Karaboga, 2005; Karaboga et al., 2007; Karaboga and Ozturk, 2009]; 4) Artificial Bee Colony and Optimized Backpropagation proposed by Bullinaria and AlYahya [2014].

A number of six neurons in the hidden layer is selected by Karaboga and Ozturk [2009] because it satisfies the ANN topology. However, Bullinaria and AlYahya [2014] states that six neurons cannot correctly classify the Digit set, so, as presented in the work, for this set, 40 neurons were applied to the hidden layer, in accordance to [Bullinaria and AlYahya, 2014].

The sample size of the experiment for each dataset was established to be 50, starting with a random population with different seeds. The stopping criterion adopted was 2000 generations except for Digits dataset was 10000 generations. Without a demand limit in the value of the weights. The parameters adopted for the KX Crossover were: 1) Population with 30 Individuals; 2) Selection tournament method; 3) Rate of crossover set to 80%; 4) Rate of mutation as 30%. These values were adopted following the rationale of [Karaboga et al., 2007; Karaboga and Ozturk, 2009]. The parameters of the other algorithms can be found in [Karaboga and Ozturk, 2009; Bullinaria and AlYahya, 2014].

https://proceedings.science/p/85564?lang=pt-br

## 7. Results

For each dataset, the algorithm was tested 50 times with the same configuration, but with different seed. Subsequently, their results are compared with the results presented in [Karaboga and Ozturk, 2009; Bullinaria and AlYahya, 2014]. For the output layer, each neuron represents a distinct class of the problem, aiding the classification. Table 2 presents the mean of Classification Error Percentage (CEP) together with the standard deviation (SD) of the tests set for each algorithm. CEP is represented as the amount of data that the ANN fails to classify.

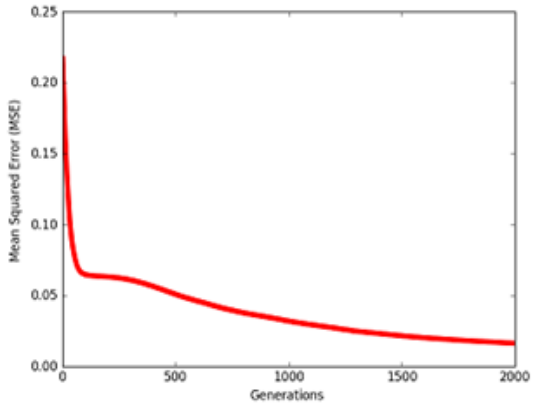
As observed in Table 2, the optimized version of Backpropagation achieved significantly better performance on Thyroid, Soybean and Digits datasets ($p < 0.01$). Comparing the KX to the canonical GA, the KX is significantly better for problems with low dimensionality, although it is significantly worse for datasets with high dimensionality ($p < 0.01$).

Considering the Heart dataset, KX was more significantly more accurate and robust ($p < 0.01$) than the others, indicating that this crossover method improves the performance of GA's for classifying small neural networks (<1000 connection weights) with few outputs.

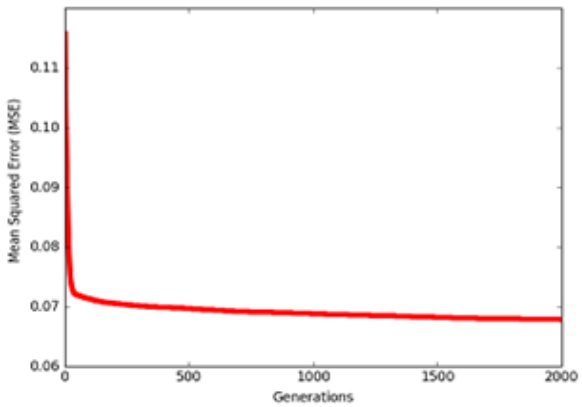
| Dataset | | KX | GA [Karaboga and Ozturk, 2009] | BP [Karaboga and Ozturk, 2009] | ABC [Karaboga and Ozturk, 2009] | Opt. ABC [Bullinaria and AlYahya, 2014] | Opt. BP [Bullinaria and AlYahya, 2014] |
|---|---|---|---|---|---|---|---|
| Thyroid | CEP | 3.64 | 7.09 | 7.26 | 6.95 | 6.14 | **2.06** |
| | SD | 0.01 | 0.03 | 0.00 | 0.01 | 0.07 | 0.21 |
| Heart | CEP | **18.41** | 20.60 | 21.44 | 19.48 | 19.13 | 20.17 |
| | SD | **0.06** | 0.68 | 0.55 | 1.41 | 1.34 | 1.14 |
| Soybean | CEP | 48.5 | 40.39 | 61.16 | 38.63 | 13.93 | **10.08** |
| | SD | 0.04 | 4.93 | 19.18 | 3.18 | 1.13 | 1.98 |
| Digits | CEP | 6.46 | - | - | - | 6.29 | **4.32** |
| | SD | 0.01 | - | - | - | 0.18 | 0.27 |

**Table 2.** Statistical Results obtained by each algorithm.

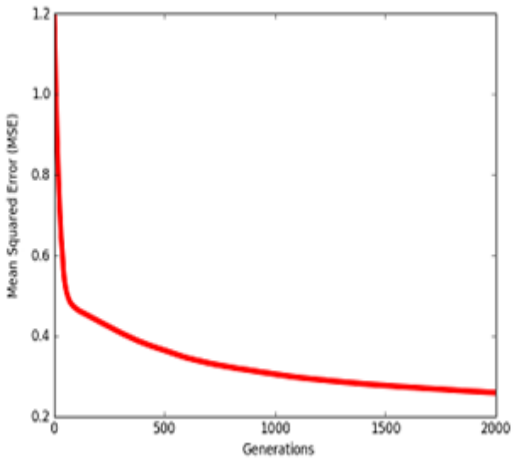
Convergence of the KX crossover Genetic Algorithm is represented in Graphs 1,2,3 and 4 where in each graph, the average of the best candidate solution of each run for each iteration is shown. The results point to a fast convergence towards the optimal in all instances.



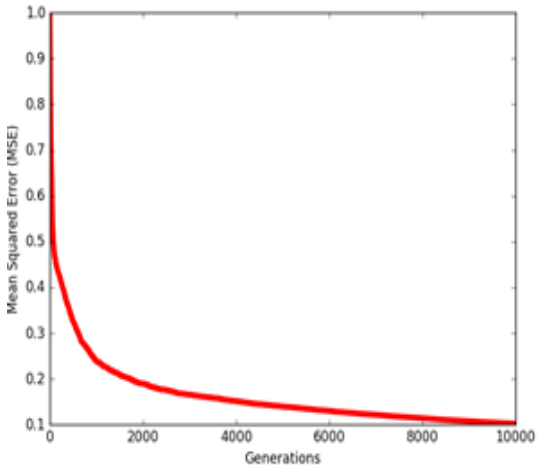

**Graph 1.** Mean of Mean Squared Error for Thyroid dataset




**Graph 2.** Mean of Mean Squared Error for Heart dataset




**Graph 3.** Mean of Mean Squared Error for Soybean dataset




**Graph 4.** Mean of Mean Squared Error for Digits dataset

## 8. Conclusion

This work had as main objective to apply a modification to the crossover step of the genetic algorithm and to verify its performance in the classification of data via Artificial Neural Networks. This modification, titled KX Crossover, had been tested in fourth ANN benchmark problems in comparison to other gradient and population-based algorithms.

Results show that the modified GA outperformed all other methods on the Heart dataset, a benchmark that has a low number of connection weights and outputs. This indicates that indeed the KX crossover guarantees faster convergence of the GA for low dimensionality instances and endorsed the concept that Evolutionary Algorithms performs better specifically for this kind of problem.

As future work, it is aimed the implementation of the algorithm, presented here, under a parallel GPGPU paradigm and the development of new tests for image classification using the MNIST dataset presented in [LeCun, 1998].

## References

Blake, C. L. (1998). UCI Repository of machine learning databases, Irvine, University of California.

Bullinaria, J. A., & AlYahya, K. (2014). Artificial bee colony training of neural networks: comparison with back-propagation. Memetic Computing, 6(3), 171-182.

Eberhart, R., Simpson, P., & Dobbins, R. (1996). Computational intelligence PC tools. Academic Press Professional, Inc.

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Lea rning. Reading, MA: Addison-Wesley.

Hagan T. M., Demuth B. H., Beale M. H., Jesús D. O. (2013) Neural Networks Design, 2nd Edition. PWS Publishing Co, Boston.

Haupt, R. L., Haupt, S. E., & Haupt, S. E. (1998). Practical genetic algorithms (Vol. 2). New York: Wiley.

Holland. J. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press.

Juang, C. F. (2004). A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 34(2), 997-1006.

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization (Vol. 200). Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.

Karaboga, D., & Ozturk, C. (2009). Neural networks training by artificial bee colony algorithm on pattern classification. Neural Network World, 19(3), 279.

Karaboga, D., Akay, B., & Ozturk, C. (2007). Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. In International conference on modeling decisions for artificial intelligence (pp. 318-329). Springer, Berlin, Heidelberg.

Karaboga, D., Akay, B., & Ozturk, C. (2007, August). Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. In International conference on modeling decisions for artificial intelligence (pp. 318-329). Springer, Berlin, Heidelberg.

LeCun, Y. (1998). The MNIST database of handwritten digits.

Morse, G., & Stanley, K. O. (2016, July). Simple evolutionary optimization can rival stochastic gradient descent in neural networks. In Proceedings of the Genetic and Evolutionary Computation Conference 2016 (pp. 477-484). ACM.

Nielsen, M. A. (2015). Neural networks and deep learning. Determination Press.

Prechelt, L. P. (1994). A Set of Neural Network Benchmark Problems and Benchmarking Rules.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1985). Learning internal representations by error propagation (No. ICS-8506). California Univ San Diego La Jolla Inst for Cognitive Science.

Weiß, G. (1994). Neural networks and evolutionary computation. I. Hybrid approaches in artificial intelligence. In Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on (pp. 268-272). IEEE.

Yasojima, E. K. K., de Oliveira, R. C. L., Teixeira, O. N., Lisbôa, R., & Mollinetti, M. (2016). Analyzing Genetic Algorithm with Game Theory and Adjusted Crossover Approach on Engineering Problems. In International Conference on Hybrid Intelligent Systems (pp. 177-188). Springer, Cham.

https://proceedings.science/p/85564?lang=pt-br