Evolving Neural Networks for Pharmaceutical Research

Crina Grosan[1], Ajith Abraham[2], Stefan Tigan[3], Tae Gyu Chang[2], Dong Hwan Kim[2]

[1]Department of Computer Science
Babes-Bolyai University, Cluj-Napoca, 3400, Romania
cgrosan@cs.ubbcluj.ro

[2]IITA Professorship Program, School of Computer Science and Engineering
Chung-Ang University, Seoul 156-756, Korea
ajith.abraham@ieee.org, tgchang@dmc.cau.ac.kr, free@dmc.cau.ac.kr

[3]University Iuliu Haţieganu, Faculty of Medicine,
Department of Biostatistics and Medical Informatics, Cluj-Napoca, Romania,
stigan@umfcluj.ro

Abstract

This article presents the application of Meta-Learning Evolutionary Artificial Neural Network (MLEANN) for a pharmaceutical research problem. Designing drugs is a current problem in the pharmaceutical research domain. By designing a drug we mean to choose some variables of drug formulation (inputs), for obtaining optimal characteristics of drug (outputs). To solve such a problem we propose an evolutionary artificial neural network and the performance is compared with a neuro-fuzzy system and an artificial neural network trained using scaled conjugate gradient algorithm. This research used the experimental data obtained from the Laboratory of Pharmaceutical Techniques of the Faculty of Pharmacy in Cluj-Napoca, Romania. Bootstrap techniques were used to generate more samples of data and the number of experimental data is reduced due to the costs and time durations of experimentations. We obtain in this way a better estimation of some drug parameters. Experiment results indicate that the proposed method is efficient.

1. Introduction

This article presents some advanced computational intelligence tools useful for modeling the situations that interfere in the process of designing drugs. By designing a drug we mean to choose some variables of drug formulation, for obtaining optimal characteristics of drug. Our application is made on a particular class of drugs, namely retard drugs. We approach this problem with a bootstrap simulation which is suitable in some particular situations.

The problem comes from the pharmaceutical research activity. It refers to a specific class of drugs that has delayed action called generically *retard drugs [2]*. The pharmaceutical experimental situation leads to a mathematical optimization problem. The pharmacist researcher must take into account several variables of formulation of the drug such as: the speed of mixing turbine, the concentration of the binder, addition speed, the proportion of talc, the proportion of lauril sulfate Na. We will call these variable of formulation *inputs* and we will denote by $X_i,\ i=\overline{1,n}$. In the case of our problem we have five inputs, X_1, X_2, X_3, X_4 *and* X_5. With those variables of formulation, for each combination took into account, the researcher obtains a variant of drug with certain characteristics. For each obtained variant are measured some parameters, called *responses* that characterize the drug: charging performance, average diameter of the pill, Carr value, Hausner value, the flow time and the brittleness. We consider these responses as *outputs* and we formalize them by denoting with $Y_j,\ j=\overline{1,m}$. For our problem we have 6 outputs denoted by Y_1, Y_2, Y_3, Y_4, Y_5 and Y_6. The costs of experimentations are high and it is necessary to devote a long time to determine all responses for each variant of drug. So, the researcher realized only 11 experiments. With the formalization proposed on top, we grouped the experimental data as depicted in Table 1.

The aim is to determine a combination of variables of formulation $(x_1, x_2, ..., x_5)$ such that the responses $(y_1, y_2, ..., y_6)$ are optimal. By optimal we mean that outputs respect some conditions. We must take into account some restraints for outputs.

2006 International Conference on Hybrid Information Technology (ICHIT'06)
0-7695-2674-8/06 $20.00 © 2006 IEEE

Variables of formulation: Inputs						Responses: Outputs					
ExpNo	**X_1**	**X_2**	**X_3**	**X_4**	**X_5**	**Y_1**	**Y_2**	**Y_3**	**Y_4**	**Y_5**	**Y_6**
1	20	2	3	5	1	84.0	973.8	4.2	1.043	7.85	1.165
2	40	2	3	0	0	71.9	1150.0	1.6	1.016	8.2	2.264
3	20	8	3	0	1	92.5	1121.4	4.2	1.044	8.83	0.700
4	40	8	3	5	0	88.1	1200.0	3.7	1.038	8.87	1.205
5	20	2	9	5	0	99.2	910.0	5.8	1.061	8.3	1.914
6	40	2	9	0	1	68.2	985.1	4.1	1.043	7.9	2.550
7	20	8	9	0	0	99.1	1010.0	5.3	1.056	9.05	1.160
8	40	8	9	5	1	83.9	925.4	5.5	1.058	8.5	1.265
9	30	5	6	2.5	0.5	85.0	1055.8	3.8	1.036	8.3	1.535
10	30	5	6	2.5	0.5	81.2	1030.0	4.1	1.042	8.37	1.490
11	30	5	6	2.5	0.5	85.0	1060.0	4.1	1.042	8.4	1.535

Table 1. Sample data showing the *inputs* and *outputs*

1. The first response, output Y_1, must be maximized, so the goal is to obtain a value as close as possible to 100 %.
2. Te second output Y_2 must not outrun some values determined by the fact it is a value representing tablet's diameter. So, the requirement is that $y_2 \in$ [800μm, 1000μm]. A value around the average 900μm is suitable.
3. The third output Y_3 has also an admissible interval for its value, [1, 20], but we must determine it as close as possible to 1.
4. The fourth response, output Y_4, has a narrower interval for its values, [1, 1.2], but it also has to be closest to 1.
5. The fifth output Y_5, representing a time quantity, must be as small as possible, but positive.
6. For the last output Y_6, the goal is to minimize it, with positive values, so the 0 value is considered desirable.

We search for the values of $X_i \;\; i=\overline{1,5}$ for which we obtain a drug formulation with optimal characteristics $Y_j \;\; j=\overline{1,6}$. Variables of formulation are chosen by the researcher from a continuous domain. Not all values are accepted. So we must consider domains of definition, real intervals, for each input variable. Accepted variation intervals for inputs, for our problem, are: $X_1 \in [0,50]$, $X_2 \in [1,8]$, $X_3 \in [3,9]$, $X_4 \in [0,5]$, and $X_5 \in [0,1]$.

2 Computational Intelligence (CI)

CI substitutes intensive computation for insight into how complicated systems work. Artificial neural networks, fuzzy inference systems, probabilistic computing, evolutionary computation etc were all shunned by classical system and control theorists. CI provides an excellent framework unifying them and even by incorporating other revolutionary methods.

Artificial Neural Networks (ANNs) were designed to mimic the characteristics of the biological neurons in the human brain and nervous system. The network "learns" by adjusting the interconnections (called weights) between layers. When the network is adequately trained, it is able to generate relevant output for a set of input data. Backpropagation (BP) is one of the most famous training algorithms for multilayer perceptrons. Basically, BP is a gradient descent technique to minimize the error E for a particular training pattern. For adjusting the weight (w_k),in the batched mode variant the descent is based on the gradient ∇E ($\frac{\delta E}{\delta w_k}$) for the total training set:

$$\Delta w_k(n) = -\varepsilon \cdot \frac{\delta E}{\delta w_k} + \alpha \cdot \Delta w_k(n-1)$$

The gradient gives the direction of error E. The parameters ε and α are the learning rate and momentum respectively. A good choice of both the parameters is required for training success and speed of the ANN.

In the Conjugate Gradient Algorithm (CGA) a search is performed along conjugate directions, which produces generally faster convergence than steepest descent directions. A search is made along the conjugate gradient direction to determine the step size, which will minimize the performance function along that line. A line search is performed to determine the optimal distance to move along the

current search direction. Then the next search direction is determined so that it is conjugate to previous search direction. The general procedure for determining the new search direction is to combine the new steepest descent direction with the previous search direction. An important feature of the CGA is that the minimization performed in one step is not partially undone by the next, as it is the case with gradient descent methods. An important drawback of CGA is the requirement of a line search, which is computationally expensive. Moller introduced the Scaled Conjugate Gradient Algorithm (SCGA) as a way of avoiding the complicated line search procedure of conventional CGA. According to the SCGA, the Hessian matrix is approximated by

$$E''(w_k)p_k = \frac{E'(w_k + \sigma_k p_k) - E'(w_k)}{\sigma_k} + \lambda_k p_k \quad (2)$$

where E' and E'' are the first and second derivative information of global error function $E(w_k)$. The other terms p_k, σ_k and λ_k represent the weights, search direction, parameter controlling the change in weight for second derivative approximation and parameter for regulating the indefiniteness of the Hessian. In order to get a good quadratic approximation of E, a mechanism to raise and lower λ_k is needed when the Hessian is positive definite. Detailed step-by-step description can be found in the literature [5][4].

We used the Meta-Learning Evolutionary Artificial Neural Network (MLEANN) framework [7] in this research. Figure 1 illustrates the general interaction mechanism with the learning mechanism of the EANN evolving at the highest level on the slowest time scale. The efficiency of evolutionary training can be improved significantly by incorporating a local search procedure into the evolution. Evolutionary algorithms are used to first locate a good region in the space and then a local search procedure is used to find a near optimal solution in this region. It is interesting to consider finding good initial weights as locating a good region in the space. Defining that the basin of attraction of a local minimum is composed of all the points, sets of weights in this case, which can converge to the local minimum through a local search algorithm, then a global minimum can easily be found by the local search algorithm if the evolutionary algorithm can locate any point, i.e., a set of initial weights, in the basin of attraction of the global minimum. In this research, back-propagation (BP) algorithm is used as the local search algorithm. All the randomly generated architecture of the initial population are trained by BP algorithm for a fixed number of epochs. The learning rate and momentum of the BP algorithm are adapted according to the problem. The basic algorithm of the proposed MLEANN framework is given below.

1. *Set t=0 and randomly generate an initial population of neural networks with architectures, node transfer functions and connection weights assigned at random.*
2. *Evaluate fitness of each ANN using BP algorithm*
3. *Based on fitness value, select parents for reproduction*
4. *Apply mutation to the parents and produce offspring (s) for next generation. Refill the population back to the defined size.*
5. *Repeat step 2*
6. *STOP when the required solution is found or number of iterations has reached the required limit.*

Architecture of the chromosome is depicted in Figure 2.

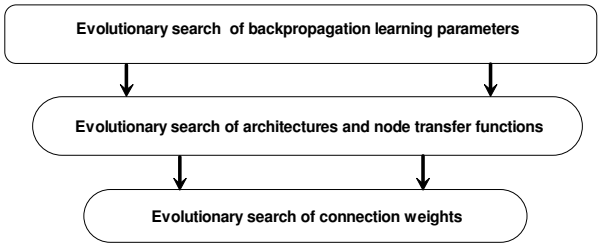

Figure 1. Interaction of various evolutionary search mechanisms in the MLEANN framework

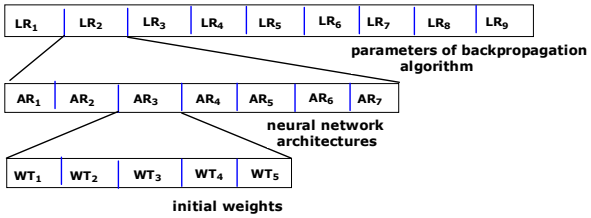

Figure 2. Chromosome representation of MLEANN

We used a special mutation operator, which decreases the mutation rate as the algorithm greedily proceeds in the search space [7]. If the allelic value x_i of the i-th gene ranges over the domain a_i and b_i the mutated gene x_i' is drawn randomly uniformly from the interval $[a_i, b_i]$.

$$x_i' = \begin{cases} x_i + \Delta(t, b_i - x_i), if\ \omega = 0 \\ x_i + \Delta(t, x_i - a_i), if\ \omega = 1 \end{cases}$$

where ω represents an unbiased coin flip $p(\omega=0) = p(\omega=1) = 0.5$, and

$$\Delta(t,x) = x\left(1 - \gamma^{\left(1 - \frac{t}{t\max}\right)^b}\right)$$

defines the mutation step, where γ is the random number from the interval $[0,1]$ and t is the current generation and t_{max} is the maximum number of generations. The function Δ computes a value in the range $[0,x]$ such that the probability of returning a number close to zero increases as the algorithm proceeds with the search. The parameter b determines the impact of time on the probability distribution Δ over $[0,x]$. Large values of b decrease the likelihood of large mutations in a small number of generations.

Performance of MLEANN is compared with a Neuro-Fuzzy (NF) framework [5][6]. We used the Adaptive Neuro Fuzzy Inference System (ANFIS) implementing a Takagi-Sugeno type FIS. We modified the ANFIS model to accommodate the multiple outputs [6]. Figure 3 depicts the 6- layered architecture of multiple output ANFIS and the readers are advised to consult [6] for technical details of ANFIS.

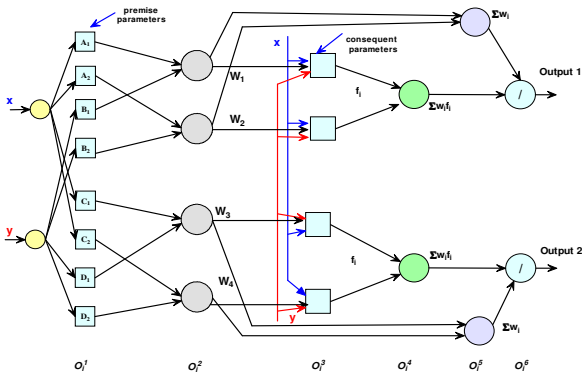

Figure 3. Architecture of ANFIS with multiple outputs

ANFIS makes use of a mixture of backpropagation to learn the premise parameters and least mean square estimation to determine the consequent parameters. A step in the learning procedure has two parts: In the first part the input patterns are propagated, and the optimal conclusion parameters are estimated by an iterative least mean square procedure, while the antecedent parameters (membership functions) are assumed to be fixed for the current cycle through the training set. In the second part the patterns are propagated again, and in this epoch, backpropagation is used to modify the antecedent parameters, while the conclusion parameters remain fixed. This procedure is then iterated.

3. Experiment Setup and Results

In this kind of pharmaceutical drug design problems the researcher has to deal with two major aspects of the experimentation:

1. The relation between inputs and outputs is unknown, so we must find a way to make the best approximation
2. Data are difficult to obtain due to the restraints of costs and time. So, a method to simulate similar data is desirable.

3.1. Bootstrap re-sampling of data

If the proposed solution is not acceptable or the subsystem described above is not compatible and the process of evaluating an optimal solution is not convergent we can conclude that the sample of data is too poor and the regression functions are not suitable to describe the links between variables [1][3]. We resort, in this case, to a resampling method that allow us to manage uncertainty. The aim is to improve our sample of data with pseudo data and to evaluate some statistical parameters. We use a bootstrap method of resampling data. We group each input variable of formulation X_i, $i = \overline{1,5}$ with each output Y_j, $j = \overline{1,6}$. We obtain 30 vectors with bivariate data (X_i, Y_j). For those vectors we apply a bivariate bootstrap resampling procedure. Finally, among the bootstrap simulated sets of X_i variables we select a combination of inputs, a set of values x_i, $i = \overline{1,5}$, that correspond to best situated values of Y_j. We mean by this to observe the re-sampling process of (X_i, Y_j) and select a proper combination of y_j, $j = \overline{1,6}$. In our application such a combination is:

$$\begin{pmatrix} y_1 = 99.1, y_2 = 910, y_3 = 1.6, \\ y_4 = 1.036, y_5 = 7.85, y_6 = 0.7 \end{pmatrix}$$

We look in the bivariate bootstrap re-sampling process and extract the input combination $(x_1, x_2, x_3, x_4, x_5)$ for which we obtain the above combination of y_j, $j = \overline{1,6}$.

From other point of view, such a re-sampling method is useful to obtain new data and to improve the function approximation of the dependence $y=f(X)$.

Output	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6
Evolutionary Artificial neural networks						
RMSE	0.011	0.019	0.0123	0.0129	0.0132	0.0101
CC	0.990	0.993	0.994	0.990	0.989	0.998
Artificial neural networks						
RMSE	0.0487	0.039	0.043	0.040	0.038	0.039
CC	0.979	0.968	0.967	0.956	0.966	0.975
MIMO Neuro-fuzzy system						
RMSE	0.0187	0.0213	0.0134	0.0145	0.0190	0.0132
CC	0.992	0.989	0.993	0.987	0.991	0.990

Table 2. Test results and performance comparison of drug design system

So, we can say that no matter what method we choose to approach such a problem, re-sampling bootstrap methods are desirable to improve the accuracy of data sets. The experimental system consists of two stages: modelling the CI models and performance evaluation. Experiments were repeated three times and the worst errors are reported. The test data is then passed through the trained models to evaluate the learning efficiency of the considered models. We have applied the MLEANN framework for drug design. For performance comparison, we used the same set of training and test data that were used for experimentations with conventional design of neural networks and a neuron-fuzzy system. Fitness value is calculated based on the RMSE achieved on the test set. In this experiment, we have considered the best-evolved neural network as the best individual of the last generation. All the genotypes were represented using real coding and the initial populations were randomly generated based on the parameters shown in Table 1. MLEANN learning (convergence) showing the best fitness values is illustrated in Figure 4.

Our preliminary experiments helped us to formulate a feedforward neural network with 1 input layer, 1 hidden layer and an output layer [5-10-6]. Input layer consists of 5 neurons corresponding to the input variables. The hidden layer consists of 10 neurons using tanh-sigmoidal activation function. Training was terminated after 2000 epochs and we achieved a Root Mean Squared Error (RMSE) of 0.0362. Figure 5 shows the convergence of SCGA during the 2000 epochs training. For training the neuro-fuzzy (NF) model, we used 2 Gaussian membership functions for each input variables and 64 rules were learned using the hybrid training method. Training was terminated after 30 epochs. For the NF model, we achieved training RMSE of 0.0218. Table 2 summarizes the training and test performances of MLEANN, neuro-fuzzy system and neural network. The individual RMSE and Correlation Coefficients (CC) for outputs Y_1, Y_2, Y_3, Y_4, Y_5 and Y_6.

Empirical results (Root Mean Squared Error – RMSE, and Correlation Coefficient – CC) using the three methods are illustrated in Table 2.

Population size	30
Maximum no of generations	25
Number of hidden nodes	5-9 hidden nodes
Activation functions	tanh *(T)*, logistic *(L)*, sigmoidal *(S)*, tanh-sigmoidal *(T*)*, log-sigmoidal *(L*)*
Output neuron	linear
Training epochs	500
Initialization of weights	+/- 0.1
Ranked based selection	0.50
Learning rate	0.15-0.01
Momentum	0.15-0.01
Elitism	5 %
Initial mutation rate	0.70

Table 3. Parameters used for evolutionary design of artificial neural networks

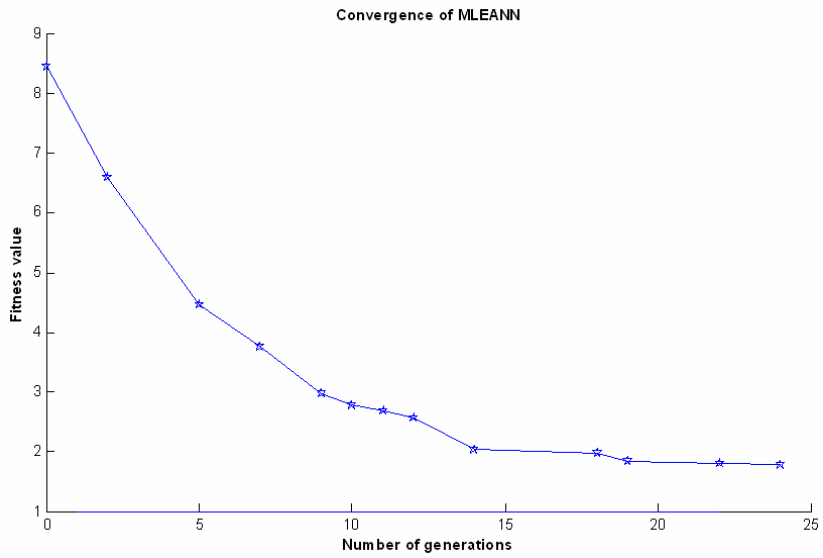

Figure 4. Convergence of MLEANN algorithm

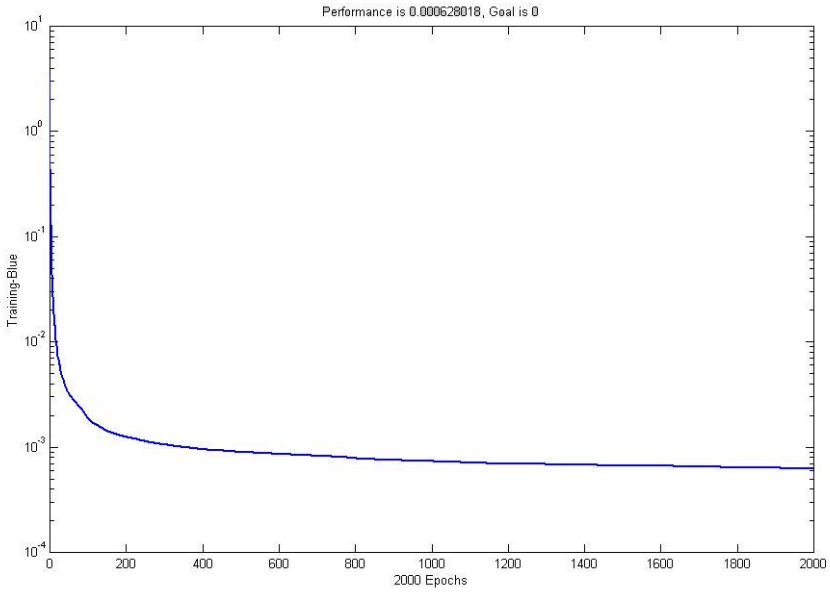

Figure 5. Convergence of SCGA training

4. Conclusions

In this paper, we have proposed a novel evolutionary artificial neural network model for drug design. Test results reveal that the proposed connectionist models are capable of modeling all the outputs accurately. Compared to a direct artificial neural network and neuro-fuzzy system, the proposed MLEANN approach performed better in terms of lower RMSE, and high correlation coefficient.

An important advantage of neuro-fuzzy system is the interpretability of the results using *if-then* rules. The proposed intelligent system might be useful for drug design researchers, companies engaged in drug business etc. Performance could have been improved by providing more training data. The most important achievement of this result is that it gives to the researcher a new starting point of experimentation in stead of making other 20 - 30

experiments and to arrive to the same conclusion as the connectionist model recommends.

Acknowledgements

This research was supported by the International Joint Research Grant of the IITA (Institute of Information Technology Assessment) foreign professor invitation program of the MIC (Ministry of Information and Communication), Korea.

Authors would also like to thank the colleagues of the Department of Maxillofacial Surgery, University of Medicine and Pharmacy, Iuliu Hatieganu Cluj-Napoca, for the initial contributions of this research

References

[1] James Carpenter, Harvey Goldstein, Jon Rasbash, *A Non-parametric bootstrap for multilevel models*, Multilevel modelling Newsletter, 11, 2-5, 1999

[2] Remus Câmpean, Augustin Prodan, *A Rating Model Applied for Designing Drugs*, Proceedings of the 12-th IASTED International Conference on Applied Simulation and Modelling, Marbella, Spain, pg 557-561, ACTA press, 2003, ISBN: 0-88986-384-9, ISSN: 1021-8181

[3] T. Hesterberg, S. Monaghan, D. S. Moore, A. Clipson, R. Epstein, *Bootstrap Methods and Permutation Tests*, W. H. Freeman and Company, New York, 2003

[4] Moller, A. F., A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning, Neural Networks. 6:525-533, 1993.

[5] Abraham A., Neuro-Fuzzy Systems: State-of-the-Art Modeling Techniques, Connectionist Models of Neurons, Learning Processes, and Artificial Intelligence, Lecture Notes in Computer Science, Jose Mira and Alberto Prieto (Eds.), Germany, Springer-Verlag, LNCS 2084: 269-276, 2001.

[6] Jang, S. R., Sun, C. T. and Mizutani, E., Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, US, Prentice Hall Inc., 1997.

[7] Abraham A., Meta-Learning Evolutionary Artificial Neural Networks, Neurocomputing Journal, Elsevier Science, Netherlands, Vol. 56c, pp. 1-38, 2004.