Expert Systems with Applications 42 (2015) 6177–6183

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

# Development of a decision support system based on neural networks and a genetic algorithm

Oleg E. Bukharov *, Dmitry P. Bogolyubov

*Moscow Institute of Electronics and Mathematics, National Research University Higher School of Economics, 34 Tallinskaya Str., Moscow, Russia*

ARTICLE INFO

*Article history:*
Available online 3 April 2015

*Keywords:*
Decision support system
DSS
Neural network
Genetic algorithm
GPGPU
CUDA

ABSTRACT

Given ever increasing information volume and complexity of engineering, social and economic systems, it has become more difficult to assess incoming data and manage such systems properly. Currently developed innovative decision support systems (DSS) aim to achieve optimum results while minimizing the risks of serious losses. The purpose of the DSS is to help the decision-maker facing the problem of huge amounts of data and ambiguous reactions of complicated systems depending on external factors. By means of accurate and profound analysis, DSSs are expected to provide the user with precisely forecasted indicators and optimal decisions.

In this paper we suggest a new DSS structure which could be used in a wide range of difficult to formalize tasks and achieve a high speed of calculation and decision-making.

We examine different approaches to determining the dependence of a target variable on input data and review the most common statistical forecasting methods. The advantages of using neural networks for this purpose are described. We suggest applying interval neural networks for calculations with underdetermined (interval) data, which makes it possible to use our DSS in a wide range of complicated tasks. We developed a corresponding learning algorithm for the interval neural networks. The advantages of using a genetic algorithm (GA) to select the most significant inputs are shown. We justify the use of general-purpose computing on graphics processing units (GPGPU) to achieve high-speed calculations with the decision support system in question. A functional diagram of the system is presented and described. The results and samples of the DSS application are demonstrated.



## 1. Introduction

Modern ideas on collecting, processing and applying knowledge are used in decision support systems (DSS), i.e. computer-based information systems designed to assist in making complicated decisions through a more profound and focused analysis of the subject area. The creation of DSS resulted from a merge of administrative information systems and database management systems.

A variety of methods are used to analyze and generate different types of decisions in DSS, e.g. search for information and knowledge in databases, situation and data analysis, precedent-based reasoning, simulation modeling, evolutionary calculations and genetic algorithms (GA), neural networks, cognitive modeling, etc.

If a DSS is based on artificial intelligence methods, it is called an intellectual DSS, or IDSS. Using a DSS, one could solve an unstructured, semistructured or even a multicriteria task.

No standard definition of the term "DSS" or its universally accepted classification are available. Researchers suggest different categorization criteria based on user-system interaction (Haettenschwiler, 1999), type of support (Power, 2007), or other approaches (Alter, 1980; Golden, Hevner, & Power, 1986; Holsapple & Whinston, 1996).

According to IDC and EMC (2011) the volume of information generated by people doubles every two years. This causes the problem of extracting the necessary information from infinite sources. Data mining is currently being actively developed as a solution to this problem. The objective of this technique is detecting the necessary information and elaborate interrelations among huge quantities of raw data.

The process of knowledge extraction at the initial stages of designing the intellectual Expert System (ES) and DSS is extremely difficult and labor-intensive, and not always successful if the databases in ill-structured subject areas contain incomplete, indistinct, polytypic or inconsistent information. The term "ill-structured" was introduced by G. Simon (1984) to designate a wide range of problems with the following characteristics: indistinct definitions,

* Corresponding author.
*E-mail addresses:* oleg_bukh.box@mail.ru (O.E. Bukharov), bogolub@mail.ru (D.P. Bogolyubov).

http://dx.doi.org/10.1016/j.eswa.2015.03.018
0957-4174/

changing terms, situations depending on a set of contexts, uncertainty, ambiguity, incompleteness, discrepancy, unreliability and diversity of initial data. Therefore, the intellectual data mining provides a promising approach to the solution of the problems described above. According to Fayyad, Piatetsky-Shapiro, and Smyth (1996), six types of tasks can be differentiated within data mining: classification, regression, clustering, summarization, dependency modeling, change and deviation detection. The latter types of tasks have recently attracted an increasing number of researchers, being the basis of Internet users' action analysis, identification of linguistic dependences in natural language texts, case history analysis to predict possible diseases, DNA sequencing, as well as trend prediction at financial exchanges. For example, from a researcher point of view, exchanges generate a large number of numerical sequences (corresponding to certain timepoints) called time series.

A time series is a sequence of statistical data on parameter value(s) collected at different timepoints. Time series data have a natural temporal order, each value characterized by time of measurement or measurement number in order. Time series significantly differ from simple data selection because analysis considers not only statistical variety and characteristics of a selection, but also the correlation between measurements and time (Shmojlova, 2002).

Time series forecasting is an essential task for many spheres of human activities, such as:

1. Medicine (e.g. forecasting reactions to various formulations and doses in a treatment course);
2. Biology (forecasting physiological and psychological features of animals and humans);
3. Sociology (forecasting social relations and demographic indicators);
4. Economics (forecasting sales volumes, exchange rates and stock prices).

## 2. Forecasting

Let us consider the most widespread statistical methods of time series forecasting (Magnus, Katyshev, & Pereseckiy, 2007) (see Table 1):

In spite of a great number of existing forecasting models, very few of them are able to find a dependence between all the factors significantly influencing the predicted value. Apart from accuracy, forecasting speed is also critical for many tasks. In such tasks, it is often necessary to strike a balance between accuracy and speed focusing on the factors making a significant impact on the predicted size. Thus, the main problems of time series forecasting are as follows:

**Table 1**
Widespread statistical methods of time series forecasting.

| Statistical method | Application scope |
|---|---|
| Extrapolation forecasting methods | A trend or long-term tendency in a time series |
| Spectral correlation data analysis with period and seasonality search | Changes repeated throughout a certain period or tendencies observed in a time series |
| Models with statistical intervention parameters | Sharp changes of a tendency resulting from an external or internal influence in a time series |
| Harmonic models or Autoregressive integrated moving average models | Constant trend fluctuations in a time series with the period unknown at the beginning of research |

1. Lack of an efficient estimation technique to evaluate the dependence between the input parameters and the predicted value;
   a. Difficulty finding the attributes with the greatest influence on the predicted value and the past period when these attributes have a significant influence on the predicted future value;
   b. The problem of determining a dependence between the identified attributes and the predicted value;
2. Application of sophisticated statistical methods requiring a high level of user skill and knowledge.

Being a model of complicated multidimensional nonlinear regression, the neural network is more accurate than the above-mentioned methods, and has a number of other advantages (Tsaregorodtsev, 2010):

1. Possibility to work with non-informative noise input signals: the neural network can reject them as useless for the task solution;
2. Possibility to work with polytypic information (continuous and discrete, qualitative and quantitative data types), which is considered to be a difficult task for statistical methods;
3. Given several outputs the neural network can solve a number of problems simultaneously;
4. There are algorithms for inverse task solution with a neural network trained to resolve a specific task. For example, it is possible to connect the new neural network inputs with the outputs of the current neural network and train the new network to produce the previous network inputs as its own outputs;
5. A neural network has fewer requirements for the qualification of its user compared to complicated statistical models capable of obtaining similar results;
6. Having initially set synaptic weights of a neural network, it is possible to recreate and check the suggested statistical models as well as improve them by network training (Haykin, 2006).

## 3. Interval neural networks

When forecasting the intervals of values, it is necessary to use interval neural networks. As professor Ishibuchi visually illustrated in his paper (Ishibuchi & Tanaka, 1993), the application of two standard neural networks for this purpose may cause forecast errors when the predicted value of the upper limit of the interval is lower than that of the lower one (see Fig. 1).

An interval neural network is a system of interconnected and interacting interval neurons. Inputs and outputs of interval neurons are intervals, each of them being a continuum set of values between the two limit values (see Fig. 2).

The interval parameters for forecasting are put into each input neuron. The value at the output of the input neuron is the same value as at its input. For all the other neurons, the interval values at the output can be calculated using the following formula:

$$Y = f\left(\sum_{i=1}^{N} w_i X_i + \phi\right),$$

where $Y$ – the output from a neuron, $w_i$ – the connection weight of the $i$th input (different for each neuron), $f$ – the activation function, $N$ – the number of neural inputs, $\phi$ – bias (different for each neuron) (Holsapple & Whinston, 1996. $X_i$ – the input to the $i$th input unit (interval). The value $Y$ is calculated by the rules of interval arithmetic:

$$A * B = [a_L, a_U] * [b_L, b_U] = [\min\{a * b | a \in [a_L, a_U], b \in [b_L, b_U]\}, \max\{a * b | a \in [a_L, a_U], b \in [b_L, b_U]\}],$$

where $*$ – any operator.

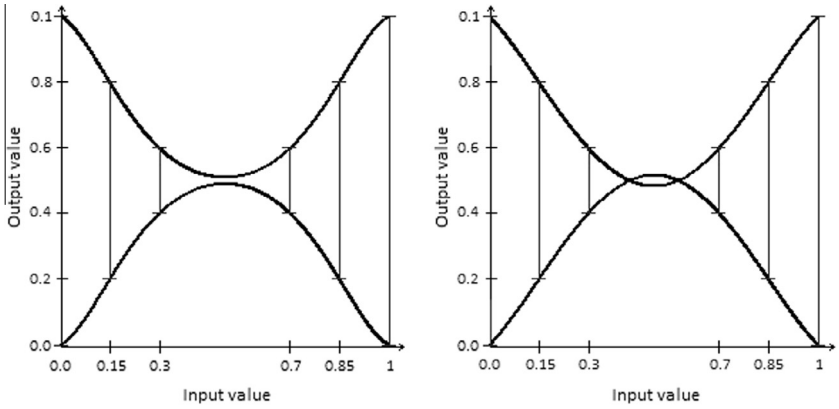


**Fig. 1.** The left graph shows the result of an interval neural network, the right graph shows the result of two standard neural networks. The vertical lines represent the output intervals of training sets.

The use of interval values allows predicting and using the following values as input parameters:

1. Standard interval data (opening/closing price, purchase/sale price);
2. The interval values making it possible to reduce the number of input values (putting in an interval from a minimum to a maximum price for a period instead of all existing rates);
3. Random variables (possible values within an interval);
4. Usual non-interval parameters (having equated the lower and upper interval limits).

We have developed a training algorithm for an interval network based on Ishibuchi and Tanaka (1991), Ishibuchi and Tanaka (1993) (generalization of backpropagation algorithm):

Let us designate the neural net inputs as $I_i = \left[I_i^L, I_i^U\right]$ and the hidden layer outputs as $H_j = f(Net_j)$, where $f$ – activation function, and $Net_j = \sum_{i=1}^{num\_inp} w_{j,i} \cdot I_i + \theta_j$; the outputs from the output layer as $O_k = f(Net_k)$, where $Net_k = \sum_{j=1}^{num\_hid} w_{k,j} \cdot H_j + \theta_k$; the target outputs $T_k = [t_k^L, t_k^U]$. The cost function (error function): $E = \frac{1}{2}\sum_{k=1}^{num\_out}\left((t_k^L - o_k^L)^2 + (t_k^U - o_k^U)^2\right)$.

The interval weights are changed according to the following rule:

$$\Delta w_{ji}(t) = \eta\left(-\frac{\partial E}{\partial w_{ji}}\right) + \beta\Delta w_{ji}(t-1), \quad \text{where}$$

$$\partial e/\partial w_{kj} = -(t_k^L - o_k^L)o_k^L(1-o_k^L)h_j^L - (t_k^U - o_k^U)o_k^U(1-o_k^U)h_j^U, \text{ if } w_{kj} \geqslant 0,$$

$$\partial e/\partial w_{kj} = -(t_k^L - o_k^L)o_k^L(1-o_k^L)h_j^U - (t_k^U - o_k^U)o_k^U(1-o_k^U)h_j^L, \text{ if } w_{kj} < 0;$$

$$\begin{aligned}\partial e/\partial w_{ji} = &-\sum_{k=1,w_{kj}\geqslant 0}^{num\_out}((t_k^L - o_k^L)o_k^L(1-o_k^L)w_{kj})h_j^L\left(1-h_j^L\right)I_i^L \\ &-\sum_{k=1,w_{kj}\geqslant 0}^{num\_out}((t_k^U - o_k^U)o_k^U(1-o_k^U)w_{kj})h_j^U\left(1-h_j^U\right)I_i^U \\ &-\sum_{k=1,w_{kj}<0}^{num\_out}((t_k^L - o_k^L)o_k^L(1-o_k^L)w_{kj})h_j^U\left(1-h_j^U\right)I_i^U \\ &-\sum_{k=1,w_{kj}<0}^{num\_out}((t_k^U - o_k^U)o_k^U(1-o_k^U)w_{kj})h_j^L\left(1-h_j^L\right)I_i^L, \\ &\text{if } w_{ji} \geqslant 0,\end{aligned}$$

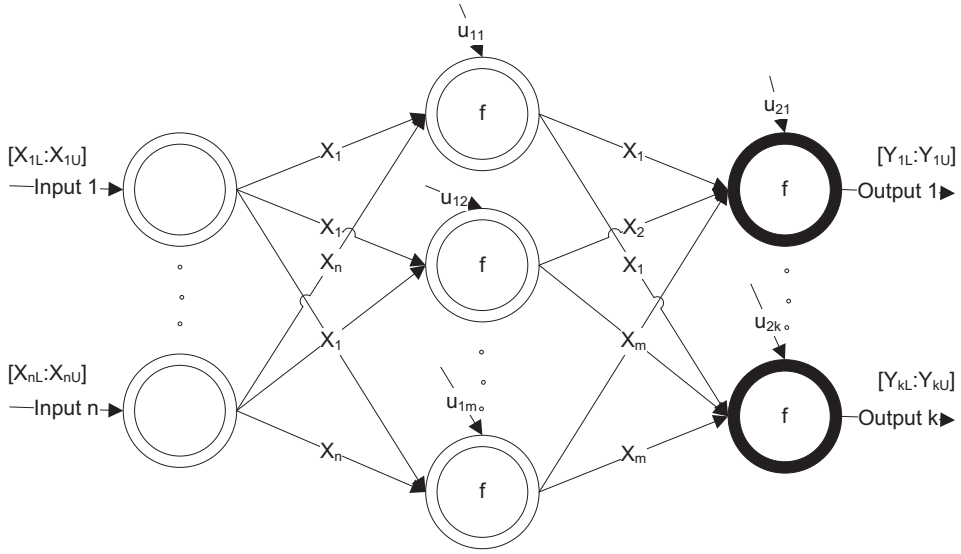


**Fig. 2.** The scheme of an interval neural network with one hidden layer ($n$ – number of input neurons, $m$ – number of hidden neurons, $k$ – number of output neurons).

$$\partial e/\partial w_{ji} = -\sum_{k=1,w_{kj}\geqslant 0}^{num\_out}\left((t_k^L - o_k^L)o_k^L(1-o_k^L)w_{kj}\right)h_j^L\left(1-h_j^L\right)I_i^U - \sum_{k=1,w_{kj}\geqslant 0}^{num\_out}\left((t_k^U - o_k^U)o_k^U(1-o_k^U)w_{kj}\right)h_j^U\left(1-h_j^U\right)I_i^L - \sum_{k=1,w_{kj}<0}^{num\_out}\left((t_k^L - o_k^L)o_k^L(1-o_k^L)w_{kj}\right)h_j^U\left(1-h_j^U\right)I_i^L - \sum_{k=1,w_{kj}<0}^{num\_out}\left((t_k^U - o_k^U)o_k^U(1-o_k^U)w_{kj}\right)h_j^L\left(1-h_j^L\right)I_i^U,$$

if $w_{ji} < 0$.

## 4. Selection of the most significant attributes

Putting all known parameters into neural network inputs will considerably slow down its operation and increase the probability of finding non-existent dependences. On the other hand, it is possible to overlook the parameters which really impact the predicted value, putting in only part of the parameters, which seem to be the most significant.

A number of algorithms have been developed for this task. The algorithm most frequently applied to neural networks is based on a successive increase in the number of significant input parameters. On the first step of the algorithm, networks with a single entrance are trained for each parameter. The parameter which results in the smallest forecast error is considered to be the most significant and is added to the list of significant parameters. Neural nets with all significant parameters plus the new one are trained on every successive step for each of the other parameters. The parameter whose network gives the smallest forecast error is added to the list. The algorithm is repeated until the addition of a new significant parameter reduces the forecast error.

Correlation analysis is not a suitable method for this task. Correlations reflect only linear dependencies between the variables, but do not reflect their functional dependencies. For example, calculating the correlation coefficient between the variables $A = \sin(x)$ and $B = \cos(x)$, we will get a value close to zero, i.e. there is no linear dependence between the variables. Nevertheless, the above variables $A$ and $B$ are obviously functionally dependent under the law $\sin^2(x) + \cos^2(x) = 1$.

A genetic algorithm was chosen as the main method of parameter identification. ***Genetic algorithms*** are adaptive search methods which are currently used to solve functional optimization tasks. They are based on the genetic processes of biological organisms: biological populations develop over the course of several generations in accordance with the laws of natural selection and the "survival of the fittest" principle discovered by Charles Darwin. Imitating this process, genetic algorithms are capable of "developing" solutions to real tasks if the latter are coded appropriately. They work with sets of "individuals", i.e. populations, each individual presenting a possible solution to the problem. Each individual is estimated by a measure of its fitness function (how close a given solution is to achieving the objectives). The most adapted individuals are capable of "reproducing" the next generation by means of a "crossover" with other individuals of the population. It helps to create new individuals combining some characteristics inherited from the parents. The least adapted individuals have a lower probability of reproduction. Therefore, their individual features will gradually disappear from the population in the course of evolution. A new population of possible solutions is reproduced this way, by selecting the best representatives of the previous generation, crossing them and producing a set of new individuals. This new generation contains a better set of characteristics inherited from the best representatives of the previous generation. Thus, from generation to generation, "good" characteristics extend to the entire population. Crossing the fittest individuals allows us to investigate the most promising areas of the search space. Finally, the population will converge to the optimum task solution.

**Table 2**
Comparative characteristics of the methods.

| Method | Linear dependency detection | Functional dependency detection | Process of obtaining the first significant results |
|---|---|---|---|
| Successive increase in significant parameters number | Yes | Yes | Slow |
| Correlation analysis | Yes | No | Fast |
| Genetic algorithm | Yes | Yes | Fast |

The initial generation of parameter sets (individuals) for a genetic algorithm is determined randomly. Thereafter, the sets are accepted as the fittest if the network trained with them gives the minimum error. The new generation of individuals is obtained by crossing the fittest individuals of the previous generation and mutation. The crossing (crossover) is performed as follows: both chromosomes are randomly divided into parts, which are later swapped. New generations are produced until the population has converged. On every step, the individuals are subject to low-probability mutation to prevent premature convergence. In case of a mutation, a separate gene parameter is replaced with another one from the general set of parameters.

In Table 2 the genetic algorithm is shown to be preferable to other methods, being able both to reveal linear and functional dependency and operate faster than the alternative techniques.

## 5. Usage of graphics processing units (GPU) for parallel calculations

The evolutionary approach to the system we have developed is manifested in the use of genetic algorithm, interval neural networks, and general-purpose graphics processing units (GPGPU).

GPU calculations involve using central processing units (CPU) together with GPUs to accelerate the calculation by means of large-scale parallelization of algorithms. This calculation method was invented more than ten years ago (Fung & Mann, 2004). It is now actively used to solve a wide range of tasks requiring fast performance of cumbersome calculations.

In spite of the fact that the cores of graphic processors are not as fast as those of central processors, the former are superior due to the number of the cores (from about 300 cores on standard graphic cards to more than 4000 cores on one of the latest ASUS products).

One of the most expensive products of the "home" range of Intel processors, Intel Core i7-975 XE of 3.33 GHz produced in 2009, has 53.3 Gflops peak productivity whereas the GPU Nvidia Tesla K10 (3072 cores) has as many as 4.58 Tflops (NVIDIA official website) due to parallel calculations. Moreover, the cost of the graphic card is much lower than that of a CPU cluster with a similar productivity.

The disadvantages of using GPU are a rather low speed of copying data from a primary storage to GPU memory, and the necessity of involving all available cores simultaneously to achieve optimum productivity.

Nvidia CUDA (Compute Unified Device Architecture) was used to implement our system. CUDA is a parallel computing platform and programming model making parallel calculations with the help of Nvidia graphic processors (NVIDIA official website) to support the GPGPU approach.

Nvidia was one of the first to present this technology with GPU of the eighth generation, i.e. G80 (GeForce 8800 GTX, 2006), and continues to support and develop it in their products.

Thus, using a combination of CPU and GPU results in achieving an unprecedented productivity with an ordinary PC by virtue of simultaneous processing of the sequential part of the code on a CPU and calculating its parallelized parts on a GPU.

## 6. Detailed description of the proposed system

By combining a genetic algorithm and an interval neural network we have produced a universal forecasting system operating within acceptable time limits. The functional diagram of the created system is presented in Fig. 3

Prior to starting the search algorithm for optimal neural network to forecast a required parameter, the user will be able to specify the following settings:

1. The name and path of the CSV file containing the time series for network learning;
2. The name and path of the CSV file containing the time series for forecasting;
3. The path for saving high-quality networks;
4. Input column numbers and the predicted value column number;
5. Input amount range;
6. Time window size range for forecasting inputs (the quantity of successive time series values to be used as a forecasting basis);
7. Time window size for the value to be predicted (the quantity of successive values to be predicted);
8. Time lag between the input window and the window of predicted values (can be negative);
9. The upper and lower limits of the values presented (for normalization);
10. Genetic algorithm population size;
11. The number of hidden neurons in the neural networks;
12. The number of neural network learning cycles;
13. The number of neural networks learning with the same set of inputs;
14. The maximal number of iterations.

The system algorithm:

1. The first step of the search algorithm is the creation of initial generations. Random sets of input parameters are generated. Each set is limited to a predefined input amount and time window size range.
2. The next step is to train a set of neural networks for each set of parameters. Learning is carried out using CUDA for parallel computing on GPU, which significantly reduces the algorithm's operating time.
3. If there is a network working better than the others (i.e. with a smaller error) among the trained networks, the parameters of this network are saved in the high-quality network pool. As soon as the first network is in the pool, the forecasting algorithm can work simultaneously with the search algorithm, choosing a network with the smallest error.
4. If the population has not converged or the maximum number of iterations has not been exceeded, the algorithm identifies sets of the best individuals with the smallest forecasting error.
5. A new generation is produced by crossover and mutations of the selected best sets.
6. The algorithm is completed if the maximum number of iterations has been exceeded or in the case of population convergence. Otherwise, the algorithm is repeated from step 2.

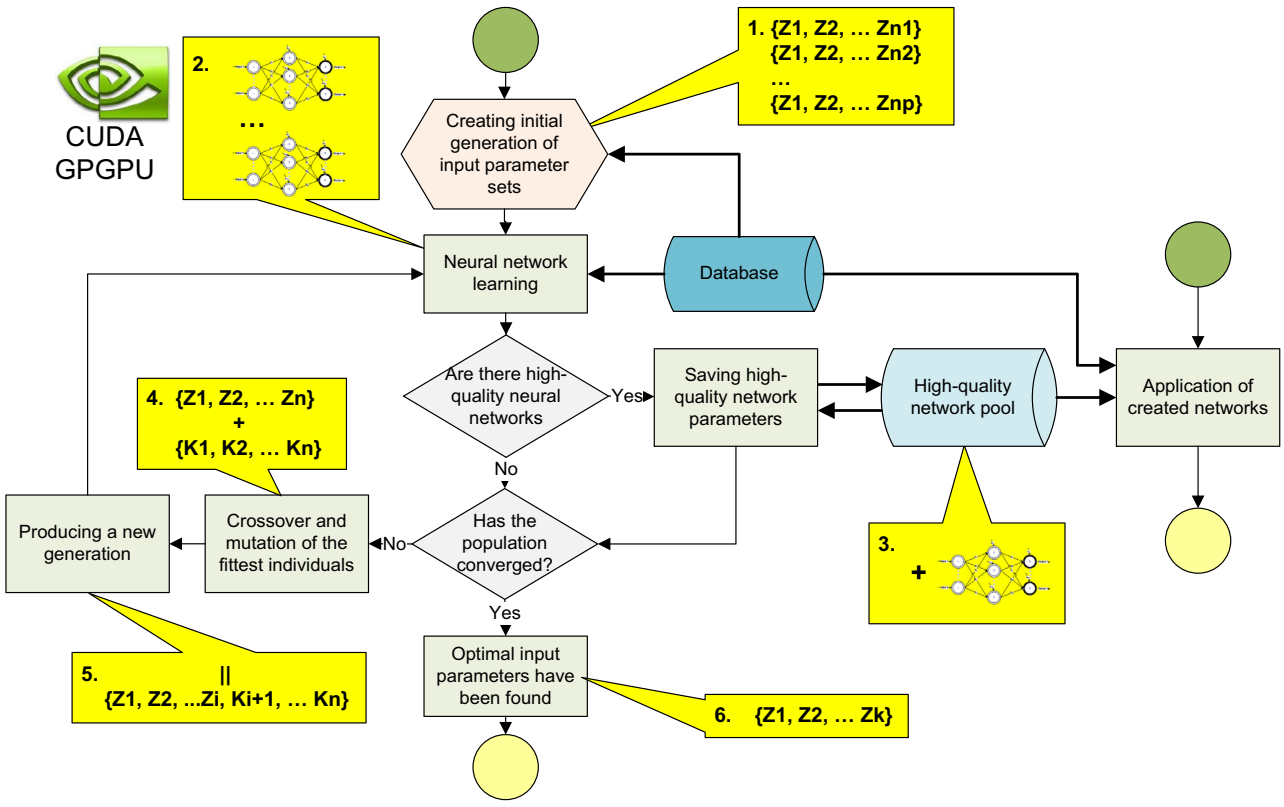


**Fig. 3.** The functional diagram of the proposed system.

**Table 3**
Running time depending on the computation type.

| Computation type | Network number | Iteration number | Number of learning cycles | Running time (s) |
|---|---|---|---|---|
| Sequential | 1 | 10 | 1000 | 0.51 |
| | 2 | 10 | 1000 | 1.07 |
| | 3 | 10 | 1000 | 1.66 |
| Simultaneous (GPGPU) | 1 | 10 | 1000 | 0.47 |
| | 2 | 10 | 1000 | 0.53 |
| | 3 | 10 | 1000 | 0.58 |

If the population has converged, the algorithm has got the optimal set of inputs and the corresponding neural network has been saved in the pool of high-quality networks.

The scheme shows that one of the key advantages of the proposed system is the opportunity to carry out forecasting as soon as the first cycle of genetic algorithm has been finished. This is possible due to replenishment of the pool with the networks with the smallest forecasting error on each step of the genetic algorithm, which results in a more precise value prediction.

The second and main advantage of the system is the parallelization of the most labor-intensive procedure, i.e. training a set of neural networks. Owing to the GPGPU model, this algorithm works significantly faster than a similar one using just a CPU.

We trained the same sets of identical neural networks using the sequential and simultaneous versions of the learning algorithm. Each network in a set has 10 input, 15 hidden and 2 output neurons.

Table 3 shows that in similar conditions the sequential algorithm application leads to a linear increase of the running time depending on the number of the leaning neural networks, whereas the increase is almost ten times slower in the simultaneous version.

## 7. Analysis of similar approaches

A number of scientists suggest using genetic algorithm to adjust connection weights and biases of neural networks (Chandwani, Agrawal, & Nagar, 2015; Wang et al., 2011).

There are more recent techniques based on adjusting not only weights and biases but also the structure of neural networks by means of genetic algorithm (Castellani, 2013; Khan, Ahmad, Khan, & Miller, 2013; Turner, Dudek, & Ritchie, 2010).

An interesting idea of neural net improvement via genetic algorithms was proposed by Bhardwaj and Tiwari (2015). Their method results in simultaneous formation of an optimal neural network structure and connection weights using genetic algorithms. They suggested genetic algorithm optimization methods to increase the convergence speed, which, however, seems to somewhat decrease the probability of finding the global extremum.

Vukicevic, Jovicic, Stojadinovic, Prelevic, and Filipovic (2014) suggested a similar idea of using genetic algorithm to optimize neural network parameters. But unlike the proposed system they did not use genetic algorithm to find an optimal set of input parameters, leaving this task for an expert.

None of the analyzed approaches seem to be able to operate with interval data or reveal the optimal input set of variables significantly affecting the target value.

## 8. Application example

A number of experiments were carried out to demonstrate the advantages of the system and the accuracy of the values predicted.

Table 4 shows the results of applying the system to weather forecasts in the Moscow region.

**Table 4**
Weather forecasting.

| Input time series | The range of input numbers | Inputs selected by the system | Mean absolute percentage error |
|---|---|---|---|
| Temperature<br>Air pressure<br>Precipitation<br>Wind | 1<br>1–2<br>1–4 | Temperature<br>TemperaturePrecipitation<br>Temperature<br>Air pressure<br>Precipitation<br>Wind | 7.68<br>7.41<br>5.93 |

Time series of daily interval values of temperature, precipitation, air pressure and wind speed were used to forecast the daily temperature intervals for the following 5 days. Every time series has 730 interval values (2 years).

The mean absolute percentage error for interval values was calculated as follows:

$$MAPE = \frac{100}{N}\sum_{1}^{N}\frac{1}{2}\left(\left|\frac{(\hat{x}_i^L - x_i^L)}{x_i^L}\right| + \left|\frac{(\hat{x}_i^U - x_i^U)}{x_i^U}\right|\right),$$

where $(\hat{x}^L; \hat{x}^U)$ is a forecasted interval value, $(x^L; x^U)$ is a real interval value, $x^L$ is the lower limit of the interval, $x^U$ is the upper limit of the interval, $i$ is the sequential number of the forecast.

Analyzing the results (Table 4), we can see that our system found the temperature to be the most significant factor influencing forecasted values. Nevertheless, the system identified all the inputs as significant when it was given the choice.

We are now actively using the developed system to forecast the basic circulation processes in the atmosphere using the information of incoming solar radiation and other geographical and weather parameters. Based on the satisfactory results of the experiments, we intend to improve the input data and achieve significant results with the proposed system.

## 9. Conclusions

Analyzing the methods used and the final tests of the system we can make the following conclusions:

1. The proposed DSS is an absolutely new model of high-speed DSS which can work with inaccurate, random or standard interval data.
2. The DSS effectively copes with complicatedly structured and dependent data. It succeeds in training interval neural nets including up to 30 inputs without a significant decrease in speed or accuracy.
3. It is reasonable to apply the developed DSS to tasks and problems that are difficult to formalize, e.g. forecasting daily exchange rate changes, social processes, climatic indicators.
4. It is also advisable to apply the system when working with interval data, e.g. inaccurate values, random variables, dynamic variables, which cannot be properly analyzed by means of standard methods.

Applying forecasting methods that rely on statistical data, we should always bear in mind that the forecast may become inaccurate or completely wrong if an unexpected new factor or unforeseen value appear.

The purpose of the proposed system is to provide real-time operation in an extreme mode (ability to start forecasting after the first algorithm iteration) and find a suboptimum method of forecasting by means of neural networks. The system can also be used to improve the quality of the existing value-predicting networks through the analysis of new data.

The developed model simultaneously achieves two objectives: finding a suboptimum set of parameters with the most influence on the investigated value and designing a neural network trained on this set. Such systems used to be extremely resource-intensive, expensive to implement, and service and time-consuming. But now it is becoming increasingly realistic and attractive thanks to the development of ideas and new technologies.

The main innovative value of our research consists in the new heuristic method of finding an optimal structure and selecting adequate parameters of interval neural network, based on the integration of the neural network and genetic algorithm.

The practical application of the above theory was development of the corresponding software which resulted in improved interval neural net learning process and increased learning speed.

1. The related areas to be studied in our further research are as follows: Increasing the variety of DSS input data by adding a fuzzy information processing module.
2. Improving DSS forecasting quality by developing new methods of interval neural network learning.
3. Achieving higher neural network learning speed based on further development of GPGPU technologies.
4. Applying the system to sea ice area forecasting.
5. Using the system to forecast financial indicators.

## References


Alter, S. L. (1980). *Decision support systems: current practice and continuing challenges*. Reading, Mass: Addison-Wesley.

Bhardwaj, A., & Tiwari, A. (2015). Breast cancer diagnosis using genetically optimized neural network model. *Expert Systems with Applications, 42*(10), 4611–4620 (accessed 14 February 2015).

Castellani, M. (2013). Evolutionary generation of neural network classifiers – An empirical comparison. *Neurocomputing, 99*(1), 214–229.

Chandwani, V., Agrawal, V., & Nagar, R. (2015). Modeling slump of ready mix concrete using genetic algorithms assisted training of artificial neural networks. *Expert Systems with Applications, 42*(2), 885–893.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. *AI Magazine, 17*(3), 37–54. Available at: <http://www.aaai.org/ojs/index.php/aimagazine/article/view/1230/1131>. (accessed 10 March 2014).

Fung, J., & Mann, S. (2004). Using multiple graphics cards as a general purpose parallel computer: Applications to computer vision. *Proceedings of the 17th International Conference on Pattern Recognition (ICPR2004)* (Vol. 1, pp. 805–808). United Kingdom: Cambridge.

Golden, B., Hevner, A., & Power, D. J. (1986). Decision insight systems: A critical evaluation. *Computers and Operations Research, 13*, 287–300. N2/3.

Haettenschwiler, P. (1999). Neues anwenderfreundliches Konzept der Entscheidungs-unterstutzung. *Gutes Entscheiden in Wirtschaft Politik und Gesellschaft* (pp. 189–208). Zurich: Hochschulverlag AG.

Haykin, S. (2006). *Neyronnye seti: polnyi kurs, 2-e izd., ispr*. [Neural networks: a comprehensive foundation, 2nd ed., Rev.] (pp. 388–389). M. Viliams.

Holsapple, C. W., & Whinston, A. B. (1996). *Decision support systems: A knowledge-based approach*. Minneapolis: West Publishing Co.

Ishibuchi, H., & Tanaka, H. (1993). An architecture of neural networks with interval weights and its application to fuzzy regression analysis. *Fuzzy Sets and Systems, 57*, 27–39. North-Holland.

Ishibuchi, H., & Tanaka, H. (1991). An extension of the BP-algorithm to interval input vectors. In: *Proc. IEEE Int. Joint Conf. on Neural Networks, Vol. 2* (pp. 1588–1593). Singapore.

Khan, M. M., Ahmad, A. M., Khan, G. M., & Miller, J. F. (2013). Fast learning neural networks using Cartesian genetic programming. *Neurocomputing, 121*(9), 274–289.

Magnus, Ja. R., Katyshev, P. K., & Pereseckiy, A. A. (2007). *Jekonometrika. Nachal'nyi kurs: Ucheb. – 8-e izd., ispr*. [Econometrics. Introductory course: Textbook. – 8th ed., Rev.] (pp. 235–279). M. Delo.

NVIDIA official website: <http://www.nvidia.com/object/tesla-servers.html> (accessed 10 March 2014).

NVIDIA official website: <https://developer.nvidia.com/what-cuda> (accessed 10 March 2014).

Power, D. J. (2007). A brief history of decision support systems. *DSSResources.COM*, version 4.0, March 10. Available at: <http://DSSResources.COM/history/dsshistory.html> (accessed 10 March 2014).

Press release: IDC, EMC (2011). Digital universe study: Extracting value from chaos. Available at: <http://www.emc.com/about/news/press/2011/20,110,628-01.htm> (accessed 24 August 2013); <http://www.emc.com/collateral/demos/microsites/emc-digital-universe-2011/index.htm> (accessed 24 August 2013).

Simon, H. A. (1984). The structure of ill-structured problems. *Developments in design methodology*. Chichester, UK: J. Wiley & Sons (pp. 317–327). Chichester, UK: J. Wiley & Sons.

Shmojlova, R. A. (2002). *Obshhaya teoriya statistiki: Uchebnik* [General Theory of Statistics: A Textbook.], M. Finansy i statistika [M. Finance and Statistics].

Tsaregorodtsev, V. G. (2010). *Preimuschestva i dostoinstva neyronnykh setey* [The benefits and advantages of neural networks]. Available at: <http://www.neuropro.ru/neu3.shtml> (accessed 10 March 2014).

Turner, S. D., Dudek, S. M., & Ritchie, M. D. (2010). Grammatical evolution of neural networks for discovering epistasis among quantitative trait loci. *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, 6023*, 86–97. Springer.

Vukicevic, A. M., Jovicic, G. R., Stojadinovic, M. M., Prelevic, R. I., & Filipovic, N. D. (2014). Evolutionary assembled neural networks for making medical decisions with minimal regret: Application for predicting advanced bladder cancer outcome. *Expert Systems with Applications, 41*(18), 8092–8100.

Wang Q., Zhang Y., Hu B., & Zhao J. (2011). Improved genetic neural network for image segmentation. In: *2011 IEEE 18th International Conference on Industrial Engineering and Engineering Management (IE&EM), Vol. 3* (pp. 1694–1698).