Knowledge Engineering

THEORETIC-EXPERIMENTAL MULTICRITERIA METHOD FOR NEURAL NETWORK CLASSIFIERS' ARCHITECTURE

Albert Voronin, Yuriy Ziatdinov, Anna Antonyuk

***Abstract**: The problem state and multicriteria optimization procedure of neural network classifier's architecture is considered. The scalar convolution of criteria with nonlinear trade-off scheme is offered as a goal function. The search methods of optimization with discrete arguments are used. The neural network classifier of texts as an example is given.*

***Keywords**: multicriteria optimization, neural nets, classifier.*

***ACM Classification Keywords**: H.1 Models and Principles – H.1.1 – Systems and Information Theory; H.4.2 – Types of Systems; C.1.3 Other Architecture Styles – Neural nets*

***Conference**: The paper is selected from XVth International Conference "Knowledge-Dialogue-Solution" KDS-2 2009, Kyiv, Ukraine, October, 2009.*

Introduction

The important version of artificial neural nets are neural network classifiers. They are used for technological and medical diagnostics, different kind of information sources classification. In general case the structure of q-layer feedforward neural network classifier is represented in Figure 1.

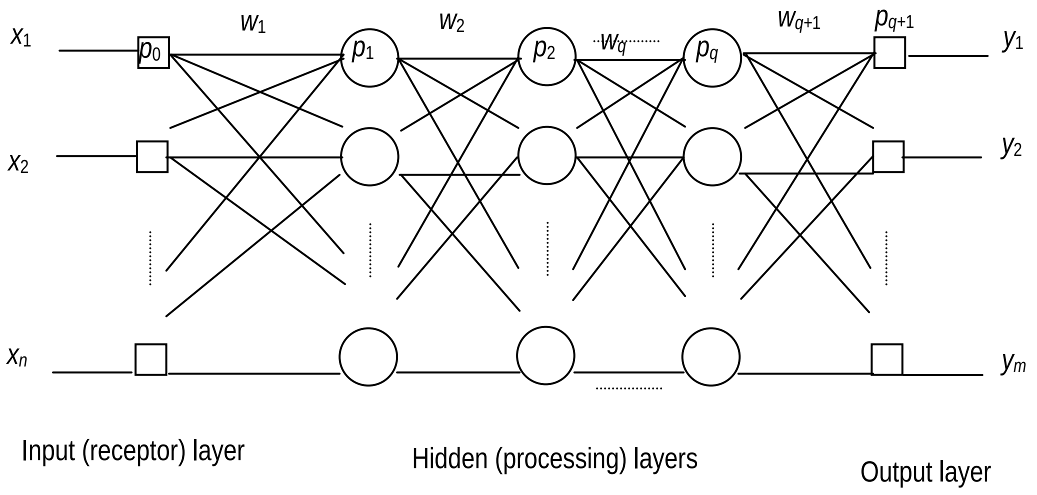

Fig. 1

Here $x_1, x_2, ..., x_n$ are features of classification object which compose the input vector $x = \{x_i\}_{i=1}^{n}; p_0 = n$ is the number of neural elements in the receptor layer; $p_1, p_2, ..., p_q$ mean number of neurons in each of hidden layers; $p_{q+1} = m$ is the number of neurons in the output layer (number of classes); $y = \{y_k\}_{k=1}^{m}$ is output vector of neural network which defines belonging of classification object to one of m classes; $w_1, w_2, ..., w_q, w_{q+1}$ - is the vector of synaptic weights of a neural network.

Let's present some necessary items from the neural networks theory [1-3]. Artificial neural network is a set of neural elements and connections between them. Each neuron has a group of synapses - unidirectional input links connected to the outputs of others neurons. Each synapse is characterized by its weight (determined at learning of a network). Neuron has current state defined as a weighted sum of its inputs: $s = \sum_{i=1}^{n} w_i x_i$. The output of neuron is a function of its state which is called the activation function: $y = f(s)$. Activation or inhibition signal through the axons (output connection of the neuron) goes to the following neuronal synapses. Activation functions can be of threshold or continuous kinds (bipolar sigmoid, gaussian, etc.). The set of all neurons of artificial neural network is divided into subsets, called layers. Layer is a set of neurons on which in every time tact enter the parallel signals received from other neurons of the network [2]. The output of the classifier is a vector of activation functions $y = \{y_k\}_{k=1}^{m}$. The index j *for* which output has maximum activity, i.e. $\max_{k \in [1,m]} y_k = y_j$, corresponds to the index of classification object class.

The number of input layer neurons is determined by the dimension of input attributes vector and can not be edited. Similarly, the number of neurons of output layer $p_{q+1} = m$ is determined by the number of classes on which the space of characters is divided, and is also a constant value. The number of processing (hidden) layers q and the number of neurons in each of them represents the conception of neural network architecture [1] and can serve as arguments (independent variables) in process of its optimization.

In this paper, we reduce the research to the case where value q is fixed and specified. Then optimization arguments of neural classifier architecture are the number of neurons in every processing layer which compose the vector of independent variables $p = \{p_j\}_{j=1}^{q}$. The quality of neural classifier operating depends on choice of architecture we do.

The problem consists in choice of such architecture when neural classifier has the best operating properties in given conditions.

Problem formulation

In general, the problem can be formally represented by the problem

$$p^* = \arg \underset{p \in P}{extr}\, Y(p), \qquad (1)$$

where $Y(p)$ is objective function; $\underset{p\in P}{extr}$ is an operator of objective extremalization function with arguments p; P is admissible domain of independent variables.

Let us make additional particular assumptions for constructive problem solution. Let associate each property of neural classifier with a quantitative characteristic $f(p)$ *which has a sense of* quality operating criterion. One of such criteria is the probability of classification error. Let us define this criterion experimentally and approximately present it as the number of classification errors $e(p)$ divided into the general, large enough number of tests N:

$$f_1(p) = \frac{e(p)}{N}. \tag{2}$$

It is supposed that with growth of neurons number in processing layers within some reasonable limits classification accuracy increases, and the value of this criterion decreases. Maximum permissible network error value should be known from physical considerations and is given as a restriction $f_1(p) \le A_1$.

The second criterion characterizes time that is needed to train the neural network with current architecture p. There is a strong correlation between that time and the total number of classifier neurons in hidden layers. So, let represent this criterion in the form of

$$f_2(p) = \sum_{k=1}^{q} p_k. \tag{3}$$

It should be noted that this criterion also characterizes signal passing time through the neural network from input to output. Criterion value increases with growth of neurons number. The maximum permissible value of the second criterion is defined by acceptable neural network training time and is presented as the restriction $f_2(p) \le A_2$.

There are other criteria for different properties of the neural classifier characteristic. In this paper we will confine ourselves to presenting just two main criteria, remembering that the proposed method allows including the other classifier properties.

Admissible domain of optimization arguments is given by the parallelepiped restriction $P = \{p \mid 0 \le p_k \le P_u, k \in [1, P_u], u \in [1, q]\}$, where P_u is the maximum number of neurons in u-th layer.

As the both of included in view criteria should be minimized (the smaller criterion, the better corresponding property of classifier), so objective function extremalization operator becomes $\underset{p\in P}{extr} = \underset{p\in P}{\min}$.

Thus, the both of criteria are contradictory, nonnegative, subject to minimization and limited. There are all the grounds to use the scalar convolution of criteria with nonlinear trade-off scheme as an objective function [4]. In unified version such convolution is represented by the formula

$$Y(p) = Y[f(p)] = \frac{A_1}{A_1 - f_1(p)} + \frac{A_2}{A_2 - f_2(p)}. \tag{4}$$

where $f(p)=\{f_r(p)\}_{r=1}^{r=2}$ is two-dimensional vector of particular criteria. Considering (2), (3) and (4), the optimization problem of neural classifier architecture for expression (1) is transformed to

$$p^* = \arg\min_{p \in P} \left[\frac{A_1}{A_1 - e(p)/N} + \frac{A_2}{A_2 - \sum\limits_{k=1}^{q} p_k} \right]. \tag{5}$$

It is easy to see that dependence $e(p)$ in formula (5) a priori is unknown and subject to experimental determination.

Method of solution

Among the optimization problems, there are such of them where arguments can acquire only discrete values according to their physical nature. Discrete values can always be reduced to an integer by special normalization. Such problems are considerably more difficult then continuous multicriteria problems and for their solutions should be applied other approaches [5].

The set of acceptable discrete values may be infinite, finite, or even consisted of only two values, 0 and 1, for example. In the first case the problem degenerates into a continuous optimization problem. To solve it the efficient and formalized algorithms and software are proposed in [4]. In the last case, the integer programming with Boolean variables, with specific methods occurs (logic synthesis of finite automaton, Rvachev's functions, etc.). From our standpoint, the most interesting and substantial is the case when the set of acceptable discrete values is not so large that problem degenerates into a continuous one, but also is not so small that problem can be solved by simple enumeration. Just this kind is the problem of nonlinear discrete (integer) programming.

Methods of discrete programming don't have such unity as methods of variational calculus have, and in most cases represent a set of particular techniques suitable for solution of particular problems. But their urgency requires their development and improvement because, as a rule, the most of important applied problems are reduced to the problems of partially or completely discrete programming. The complexity of solving discrete (integer) programming problems increases in the case of multicriteria problem.

In the case when components of possible multicriteria problems solutions can acquire only discrete values $p_k^{(P_u)}, k \in [1, P_u], u \in [1, q]$, the scalar convolution of criteria with nonlinear trade-off scheme $Y(p)$ is the lattice function defined on the discrete set P. Optimization of lattice objective function build on nonlinear trade-off scheme is reduced to the nonlinear programming problem with discrete (integer) arguments, which solution, as noted above, is difficult enough.

To solve this problem, we assume that under the discrete set P there is an auxiliary area of continuous arguments $p_c \in P_c$, which contains all discrete points $p_k^{(P_u)}$ and all continuous space between them. In the area P_c the continuous function $Y(p_c)$ is defined, which coincides with the lattice function $Y(p)$ in points $p_k^{(P_u)}$.

This assumption allows obtaining analytical solution, if in the expression (5) the dependence $e(p)$ is specified in the regression model form, for example.

Then it is possible to use the necessary condition of the function's minimum: $\frac{\partial Y(p_c)}{\partial p_c} = 0$. The solution of this equations system gives the trade-off - optimal continuous point $p_c *$. The last step of the algorithm is searching on P the discrete point nearest to $p_c *$, which will be the required discrete solution $p *$. Unfortunately, in our case specifying of the analytic dependencies is very difficult or even impossible.

As a basic we consider the case when functions $e(p)$ and therefore $Y(p)$ are unknown, but it is possible to define $Y(p)$ function's values at the points $p_k^{(P_u)}$ by measurement or calculation. Then we can organize a nature or calculating experiment, which in result will realize the search movement to the required trade-off - optimal discrete point $p *$.

There are different approaches to the search procedure organization, which should give the sequence of improving solutions. One of them is the discrete analogue of the Nelder-Mead simplex-planning method (the method of deformed polyhedron) [4]. This is the modification of gradient methods, which is very often and successfully applied in practice. The second is the non-local (dual) approach [4], which is often more efficient than the gradient methods.

As the search procedures use local or nonlocal models of continuous function $Y(p_c)$, so for called above variants exist the general necessity of searching the discrete point p_d on P which is the nearest to the continuous solution p_c at the current or final iteration. If the number of hidden layers q is not great, then the solution of this problem is not difficult (a simple rounding to the integer value). With multilayer classifiers we recommend to use the following algorithmic technique. At the point p_c is placed the center of hypersphere, which diameter increases from zero until the surface of the sphere will not touch the nearest discrete point, which thereby is identified as p_d. There are various software implementation of this algorithm. There are neural classifiers of various types and purposes.

Multicriteria optimization of neural network texts classifier

As an example, let us consider in general terms the optimization problem of neural network text classifier architecture. Text-classification system [3] consists of two main parts: the frequency analyzer with system dictionary and neural network classifier itself (Fig. 2).

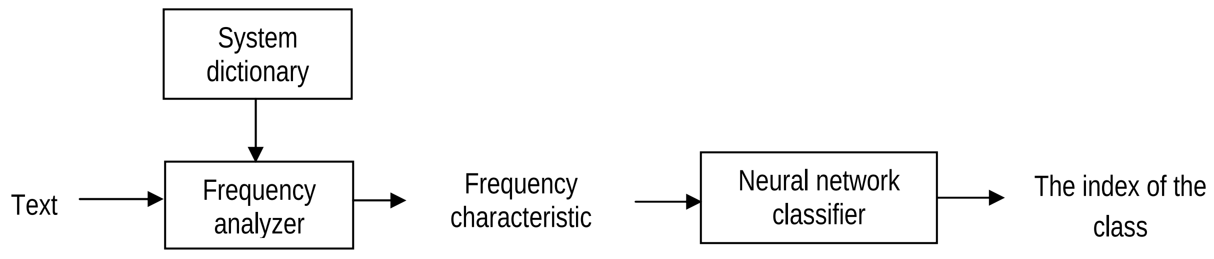

Fig.2

The text enters the input of the system, the output conforms the index of subject to which the text refers (business, politics, medicine, sport, spam, etc.). Before we proceed to optimize the neural network classifier architecture, it is necessary to perform the following steps:

1. Identifying m classes, with which the system will operate.

2. Selecting suitable training texts $t_k, k \in [1, m]$ and verifying (test) texts $t_l, l \in [1, L], L \geq m$.

3. From the set of training texts in special way words $v_i, i \in [1, n]$ are extracted and the system dictionary V is formed.

4. The frequency analyzer determines for each word v_i from the system dictionary V its frequency of occurrence x_i in the given text t_k. Frequency characteristic is the vector $x = \{x_i\}_{i=1}^{n}$ of text attributes, which dimension is equal to the number of words in the system dictionary $v_i \in V$.

After obtaining of the training texts frequency analyzer results, we can begin to train the neural classifier with some architecture $p = \{p_j\}_{j=1}^{q}$. The training process of neural network consists in specifying of such its weighting coefficients $w_1, w_2, ..., w_q, w_{q+1}$ where the maximum network error at the training texts for this architecture does not exceed the maximum permissible value. Specific learning algorithms are not considered.

Now we can proceed directly to the vector optimization procedure. To optimize the neural network classifier architecture let us use the search method of simplex-planning. Suppose, for specificity that the number of processing layers q = 2. Then the idea of the method in continuous form can be illustrated by Fig.3.

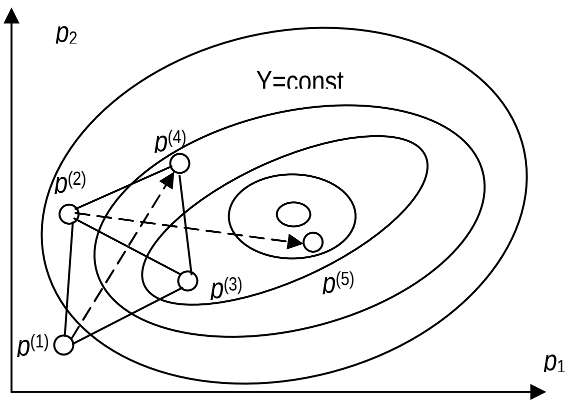

Fig.3

On the arguments plane $p_1 - p_2$ in some starting area we construct the initial regular simplex, which in two-dimensional case is represented by isosceles triangle with apexes $p^{(1)}, p^{(2)}, p^{(3)}$. For each of three simplex architectures we realize the process of the classifier learning and feed to inputs a series of test texts t_l. In each simplex apex we determine the number of classification errors $e^{(1)}, e^{(2)}, e^{(3)}$ with total number of tests $N = L$. By formula (2) we get criteria $f_1^{(1)}, f_1^{(2)}, f_1^{(3)}$. By formula (3) criteria

$f_2^{(1)}, f_2^{(2)}, f_2^{(3)}$ are defined. Formula (4) that serves in this case, as not objective, but as the evaluation function for our example is given by

$$Y(p_1, p_2) = \frac{A_1}{A_1 - e(p_1, p_2)/L} + \frac{A_2}{A_2 - p_1 - p_2} \tag{6}$$

It provides the values of the scalar convolutions $Y^{(1)}, Y^{(2)}, Y^{(3)}$ for start simplex architectures. Comparing these values between each other, we find that one of them, $Y^{(1)}$, for example, is bigger (i.e. worse) than others. Most probably it could be predicated that the architecture $p^{(4)}$ obtained by mirror reflection of the worst in base simplex point $p^{(1)}$ relatively to the center of opposite bound, would be better. By all calculations for architecture $p^{(4)}$ let us form the new simplex with apexes $p^{(2)}$, $p^{(3)}$ and $p^{(4)}$. Comparing the values $Y^{(2)}, Y^{(3)}, Y^{(4)}$, we find that one of the points, $p^{(2)}$, for example, is worse than others in the second simplex. By reflecting of this point relatively to the center of the second simplex opposite bound, we get the architecture $p^{(5)}$, etc., until we get the architecture p^* that corresponds to the objective function minimum.

This is only the illustration of the simplex-planning method idea. In fact, this method in the Nelder-Mead modifying provides simplexes adaptation to the topography of the objective function by means of the polyhedrons deformation, it has well-developed algorithms and software. In addition, we must not forget that we have the case of optimization with integer arguments, that dictates the necessity of nearest discrete solution p_d searching for every obtained continuous solution p_c.

The second, non-local search method is rather complicated in realization, but it is usually more effective [4, 5]. The method is based on the iterative construction of the non-local model $Y(p)$ «floated» together with the system of changing base points and refined by experimental results. The set of control points is compressed and constricted to the required extremum point (« shagreen leather »). At each iteration at the same time and interdependent is realized as our conception about the objective function at extremum region improvement, so the definition of such arguments extremum estimation, which is adequate to the level of these representations at the given iteration. Therefore, the non-local optimization method refers to the dual class and can be named as the method of dual programming.

Both of search methods provide the series of experiments execution. Obtained herewith experimental data can be used to build analytical regression models of particular criterion $f_1(p) = e(p)/L$. Using these models, we can realize not search, but analytical vector optimization of the other same type neural network classifiers architecture. If it will prove to be difficult, then a search procedure is executed, and with not nature, but computational experiment, that is much easier.

Solving the problem of the regression models construction, we must specify the type of approximating dependence, known accurate within unknown regression coefficients. Analysis of the problem leads to the assumption that with sufficient for practice accuracy, we can limit ourselves by the linear regression:

$$f_1(p_1, p_2) \approx (a_1 p_1 + a_2 p_2)/L, \tag{7}$$

where a_1, a_2 are regression coefficients, determined from experimental data by least-squares procedure. A linear regression model is checked for the adequacy by mathematical statistics methods. If it is necessary, the model can be complicated.

Considered methods provide the start of search procedure from the architecture, which in decision-maker's opinion is close enough to the optimal point. If in the search procedure occurs increasing of neurons number in processing layers, the neural networks theory [1] characterizes this approach as the constructive. If the start number of neurons is superfluous the approach is as called destructive (The Rodin principle: to model a sculpture, you need to take a block of marble and remove from it unnecessary).

The implementation of stated in the paper stages of vector optimization provides such neural network classifier architecture, which systematically correlates contradictory criteria of its functioning effectiveness.

Acknowledgements

The paper is published with financial support by the project ITHEA XXI of the Institute of Information Theories and Applications FOI ITHEA Bulgaria www.ithea.org and the Association of Developers and Users of Intelligent Systems ADUIS Ukraine www.aduis.com.ua.

Bibliography

1. E.V. Bodyansky, O.G. Rudenko, Artificial neural networks: architectures, training and application [in Russian], TELETEH, Kharkov: (2004).
2. V.A. Golovko, Neural networks: training, organization and application [in Russian], IPRZHR, Moscow (2001).
3. V.S. Borisov, "Self-training texts classifier in natural language", Cybernetics and system analysis, No. 3., 169-176 (2007).
4. A.N.Voronin, J.K. Ziatdinov, A.I. Kozlov, Vector optimization of dynamical systems [in Russian], Tehnika, Kiev (1999).
5. A.N.Voronin, P.D Mosorin., A.G Yasinsky, "Multicriteria problems of optimization with discrete arguments", in: Automation, 2000. International conference of automatic management: Works. -Vol.1, 75-78, Lvov (2000).

Authors' Information

Albert N. Voronin - National Aviation University, faculty of Computer Information Technologies, Dr.Sci.Eng., professor, 03058, Kiev-58, Kosmonavt Komarov avenue, 1, Ukraine, e-mail: alnv@voliacable.com

Yuriy K. Ziatdinov - National Aviation University, faculty of Computer Information Technologies, Dr.Sci.Eng., professor, Head of Chair; 03058, Kiev-58, Kosmonavt Komarov avenue, 1, Ukraine, e-mail: oberst@nau.edu.ua

Anna A. Antonyuk - National Aviation University, faculty of Computer Information Technologies, post graduate student, 03058, Kiev-58, Kosmonavt Komarov avenue, 1, Ukraine, e-mail: niuriel@mail.ru