International Journal of Information Technology and Business Management
29th December 2016. Vol.55 No.1

ISSN 2304-0777 www.jitbm.com

STRUCTURE-CONTROLLABLE NEURAL NETWORKS FOR SOLVING GENERALIZED INVERSES

Dian-Li Lin[1], Jia Peng[2], Sun Zhou[1,*]

[1]Department of Automation, Xiamen University, China

[2]Department of Information Management, Huaqiao University, China

Email: zhousun@xmu.edu.cn

Abstract

Solving problem of matrix calculation with Structure-Controllable Neural Network (SCNN) has advantages including generality, forward calculation, parallel structure, etc. and therefore is suitable to be realized by hardware. In this paper, a definition of Generalized Inverse matrix is introduced. On the basis of SCNN and BP learning algorithm, a two-layer neural network model and the corresponding algorithm are proposed to solve many kinds of generalized inverses. The algorithm performance is proved satisfactory by simulation tests.

Keywords: generalized inverse, neural network, back-propagation

1 INTRODCUTION

Matrix calculation is the fundamental problem in most scientific and engineering calculation including calculation of inverse of a normal square matrix, generalized inverse matrix, LU decomposition of a matrix, eigenvalue and eigen vector, etc. (Wang, Wei, Qiao, 2006). In many fields including practical engineering calculation such as engineering control, optimization, etc., it plays an important role. Over the past years many algorithms were put forward for this kind of problem. However, for all that these traditional methods were successfully employed in solving many practical problems, there still remain defects as follows in the methods:

(i) They cannot solve problems about abnormal matrices.

(ii) They do not give a general framework for matrix algebra problems' solving; they just provide one algorithm according to one specific problem.

(iii) In the main, they are based on serial computation, and so it is difficult to make them turn parallel.

A kind of neural network model advances a new way to solve the above problems. That is multi-layered forward Structure-Controllable Neural Network (SCNN) model, which has advantages as follows:

(i) There is no division operation, therefore the problem of dividing by zero is eliminated, so it is suitable for calculation of abnormal matrix and algebra equation.

(ii) The basic idea is general, thereby it is suitable for most matrix algebra problems.

International Journal of Information Technology and Business Management
29th December 2016. Vol.55 No.1

ISSN 2304-0777 www.jitbm.com

(iii) The structure is parallel, thus there is the possibility to parallelly process data with multi-CPU.

Based on these above features and urgent demand of engineering computation, increasing attention has been given to method of solving matrix algebra problems with SCNN. Some people ever suggested that BP (Back Propagation) learning algorithm be employed to solve the inverse of a normal square matrix by single-layer neural network (Zhao, 2015, Jiao, 2003). But the concrete algorithm was not given, and when the matrix is abnormal or a rectangle, this method cannot work. Furthermore, there are many kinds of definitions of inverse of a rectangle matrix, and methods for solving inverse matrices under different definitions are not alike, therefore to find the inverse becomes more complicated. In this paper, we shall investigate this problem, the *Penrose-Moore* definition of *Generalized Inverse* is introduced, and on the basis of SCNN and BP learning algorithm, we shall develop a new way that is able to find the solution of inverse of a variety of matrices. Two-layer parallel neural network models that correspond to each equation in the Penrose-Moore definition and the concrete algorithm are designed to solve many kinds of generalized inverse matrices, and the 4 error functions needed in learning with back propagation are also derived. The simulation test with computer program is proved successful. This approach can be generally applied, and the computation precision is enough.

2 BRIEF INTRODUCTION OF SCNN AND BP ALGORITHM

2.1 Brief Introduction of SCNN

SCNN that is used to solve matrix algebra problems is a kind of multi-layered forward neural network whose structure is controllable. It has two levels:

(i) The first level is RAM network consisting of random memory units.

(ii) The second level is a multi-layered neural network in which each neuron is a unit with many inputs and a single output to calculate weighted sum, and the weights are trained by learning algorithm of neural network (e.g. BP algorithm), and are controlled by RAM units of the first level.

While calculating, first, the input vector and ideal output vector, i.e. pairs of input-output patterns, are chosen. Then, weighted sum of each input vector of the input layer is calculated, i.e. outputs of the next layer's neurons are worked out. If the actual output matches the ideal output, calculation should come to the end; otherwise the connection weights should be modified by learning algorithm under restriction of the first network. This course does not end until the actual output matches the ideal output. Each weight has its 'switch'. When a switch is on, the according connection weight can be modified by learning algorithm. States of these switches are controlled by RAM units of RAM network. The second level of the network is controlled by proper selection of initial value of the weights. When all RAM units are 'on', the corresponding second level of the network is right the typical BP network. The below discussion explains that the neural network approach of self-adaptable SCNN is to look on matrix algebra problems as a special problem of pattern-identification, and to match expected pattern by a specific SCNN network.

2.2 Brief Introduction of BP Learning Algorithm

In BP network, error function is defined as:

$$E_i = \frac{1}{2}\sum\left(y_j - \hat{y}_j\right)^2$$

and

$$E = \sum E_i$$

The learning algorithm is:

$$\Delta w = \eta\left(-\frac{\partial E}{\partial w}\right)$$

Moreover, nonlinear functions of all neuron nodes are the same, generally as Sigmoid function; and the learning step η remains the same in the whole learning course, commonly $0< \eta <1$.

3 SOLVE GENERALIZED INVERSES WITH SCNN

3.1 Definition of Generalized Inverse Matrix

International Journal of Information Technology and Business Management
29th December 2016. Vol.55 No.1

ISSN 2304-0777 www.jitbm.com

First the Penrose-Moore definition of generalized inverse matrix is introduced as:

Definition 3.1 Given $A = (a_{ij}) \in C^{m\times n}$, if $n \times m$-ranked matrix G satisfies all or part of the following Penrose-Moore equations (Li, 1999):

$$AGA = A \quad (1)$$

$$GAG = G \quad (2)$$

$$(GA)^H = GA \quad (3)$$

$$(AG)^H = AG \quad (4)$$

G is called for the *Generalized Inverse Matrix* of A. H is the marker of Hermite Matrix. B^H denotes the transposition matrix of B when B is a real number matrix.

As this definition has its strict mathematical evidence, it has been wide accepted and used in recent years. According to this definition, generalized inverse matrix can be classified into $C_4^1 + C_4^2 + C_4^3 + C_4^4 = 15$ kinds.

If G is a generalized inverse matrix that satisfies the i^{th} equation, it is written as $G = A^{(i)}$ (i = 1,2,3,4); If G is a generalized inverse matrix that satisfies the i^{th} and the j^{th} equations, it is written as $G = A^{(i,j)}$; and a generalized inverse matrix that is able to satisfy all the four equations is denoted as A^+. Among the 15 kinds of generalized inverse matrix, only A^+ exists and is solitary.

3.2 Solve Generalized inverses with SCNN

Take the solution of 5 kinds of generalized inverse matrices $A^{(1)}$, $A^{(2)}$, $A^{(3)}$, $A^{(4)}$, $A^{(3,4)}$ for examples, the concrete neural network approach to find the solution is as follows.

A. Generalized Inverse Matrix $A^{(1)}$ that Satisfies Equation (1) $AGA = A$

This kind of generalized inverse matrix $A^{(1)}$ is derived from the problem of solution of linear equation group $x = A^{-1}b$.

Consider a two-layer neural network is as Figure 1.

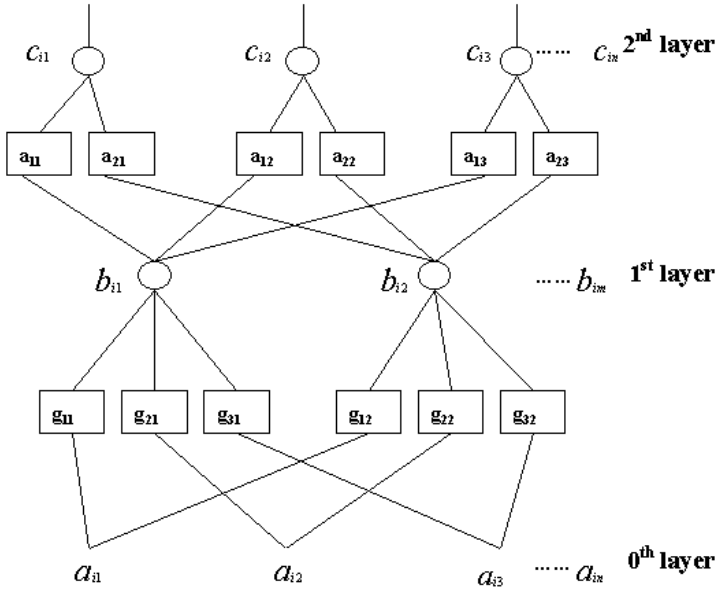

Figure 1 Two-layer neural network for finding a generalized inverse matrix that satisfies Equation (1) ($m = 2$, $n = 3$)

This neural network model embodies SCNN's characteristics that the structure is controllable, and so this model is different from typical BP network of which all neurons are interconnected.

Problem to satisfy $AGA=A$ may be looked upon as a special problem of pattern-identification with neural network. Constantly readjust elements' value

International Journal of Information Technology and Business Management
29th December 2016. Vol.55 No.1

ISSN 2304-0777 www.jitbm.com

of matrix G, so that elements of AGA might match corresponding ones of the object matrix G, thereby

$$\sum_{p=1}^{m}\left(\sum_{k=1}^{n} a_{ik}\, g_{kp}\right) a_{pj} = a_{ij}$$

where $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$. at this time weight matrix G is right the expected inverse matrix.

In Figure 1, assume that a row vector of input matrix of the network is a^i, and weight matrices from the 0th to the 1st layer, from the 1st to the 2nd layer are respectively G, A. Model of each neuron is a unit to calculate weighted sum, not Sigmoid nonlinear function in common BP algorithm. Therefore, output of the 1st layer is $a^i G$; output of the 2nd layer, i.e. output of the network, is $a^i GA$. Input all m rows to seek a generalized inverse matrix $G = (g_{ij})$ in order that the output could be the corresponding row of matrix $A = (a_{ij})$, which is the object signal. Constantly input each row of A, corresponding to different patterns, in order that all output rows could be corresponding rows of A. At this time the weight matrix G is right the expected generalized inverse matrix.

This procedure is very analogous with neural network batch-processing to identify patterns, only that neurons are units doing linear processing and how weights should be distributed has specific relation to the matrix elements' arrangement way.

Hence, to find the solution of inverse matrix that satisfies $AGA=A$, the pattern matching procedure is as follows:

(i) Choose input matrix A and corresponding object output matrix A. Here each row of A and the corresponding row of A form a pair of input-output pattern.

(ii) About the input matrix A, calculate AGA, denote the product matrix as $B = (b_{ij})$, thereby

$$b_{ij} = \sum_{p=1}^{m}\left(\sum_{k=1}^{n} a_{ik}\, g_{kp}\right) a_{pj}$$

where $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n$.

(iii) Calculate

$$E = \sum E_i = \frac{1}{2}\sum_i \sum_j \left(a_{ij} - b_{ij}\right)^2 \qquad (5)$$

If $E<\varepsilon$ (ε is given calculation precision), the matching process ends; otherwise, continue.

(iv) Value of element g_{ij} of weight matrix G is modified according to E by learning algorithm. g_{ij} changes according to update-along-gradient rule, i.e.

$$\Delta g_{ij} = \eta\left(-\frac{\partial E}{\partial g_{ij}}\right)$$

In the case of Equation (5), it is derived as

$$\frac{\partial E}{\partial g_{ij}} = \sum_{p=1}^{m}\sum_{q=1}^{n}\left\{\left[\sum_{u=1}^{m}\left(\left(\sum_{v=1}^{n} a_{pv}\, g_{vu}\right) a_{uq}\right) - a_{pq}\right] a_{pi}\, a_{jq}\right\} \qquad (6)$$

(v) Return to (ii), continue.

In actual simulation with computer program, it is found that the initial values of learning rate η and weight matrix G's elements have direct relation to whether the algorithm is able to converge, and whether variables' value is to be out of bounds, etc. Therefore, some effective practices are employed. For example, about learning rate η, in iteration for the $(t + 1)$ time, there is

$$g_{ij}(t+1) = g_{ij}(t) + \Delta g_{ij}(t+1)$$

where $\Delta g_{ij}(t+1) = \eta(t+1)\cdot\left(-\frac{\partial E}{\partial g_{ij}}\right)$. It is different from common BP algorithm in which η remains the same that here let $\eta(t+1) = 2^{\mathrm{sgn}\left[\Delta g_{ij}(t-1)\cdot\Delta g_{ij}(t)\right]}$, thereby iteration could be faster and more effective.

Here is an example.

E1. $A = \begin{pmatrix} 4.000000 & 1.000000 & 0.000000 \\ 3.000000 & 0.000000 & 3.000000 \end{pmatrix}$.

This is a rectangle matrix whose inverse matrix under traditional definition does not exist. Now the generalized inverse matrix G that satisfies equation (1) is to be found. It is demanded that $\varepsilon<10^{-6}$.

With the proposed method G can be got as

$$G = \begin{pmatrix} 0.192773 & -0.017767 \\ 0.228906 & 0.071067 \\ -0.192773 & 0.351100 \end{pmatrix}.$$

International Journal of Information Technology and Business Management
29th December 2016. Vol.55 No.1

ISSN 2304-0777 www.jitbm.com

AGA is given below with a view to check it against the object matrix so as to test the result:

$$AGA = \begin{pmatrix} 3.999999 & 0.999999 & 0.000003 \\ 3.000001 & 0.000001 & 2.999997 \end{pmatrix}.$$

It is shown that under given precision demand *AGA* = *A*, thereby the structure-controllable neural network approach for solving generalized inverse matrix is proved correct and effective.

B. Generalized Inverse Matrix $A^{(2)}$ that Satisfies Equation (2) $GAG = G$

The network structure is similar to Section A. Take matrix *G* as input matrix and the weight matrix of the 2nd layer; *A* is the weight matrix of the 1st layer, then actual output matrix of the network is *GAG*. *G* is taken as the object matrix at the same time. Calculation method is the same as the above.

Here,

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}\left[\sum_{u=1}^{n}\left(\sum_{k=1}^{m} g_{ik}\, a_{ku}\right) g_{uj} - g_{ij}\right]^2 \qquad (7)$$

It is derived that

$$\frac{\partial E}{\partial g_{ij}} = \sum_{\substack{u=1 \\ u\neq i}}^{n}\left\{\left[\sum_{v=1}^{n}\left(\left(\sum_{k=1}^{m} g_{uk}\, a_{kv}\right) g_{vj}\right) - g_{uj}\right]\left(\sum_{k=1}^{m} g_{uk}\, a_{ki}\right)\right\}$$

$$+ \sum_{\substack{u=1 \\ u\neq j}}^{m}\left\{\left[\sum_{v=1}^{n}\left(\left(\sum_{k=1}^{m} g_{ik}\, a_{kv}\right) g_{vu}\right) - g_{iu}\right]\left(\sum_{v=1}^{n} g_{vu}\, a_{jv}\right)\right\}$$

$$+ \left[\sum_{v=1}^{n}\left(\left(\sum_{k=1}^{m} g_{ik}\, a_{kv}\right) g_{vj}\right) - g_{ij}\right]\left(\sum_{\substack{v=1 \\ v\neq i}}^{n} g_{vj}\, a_{jv} + 2\, g_{ij}\, a_{ji} - 1\right)$$

(8)

E2. $$A = \begin{pmatrix} 1.000000 & 2.000000 & 2.000000 \\ 1.000000 & 2.000000 & 2.000000 \\ 3.000000 & 0.000000 & 3.000000 \end{pmatrix}.$$

Its determinant det*A* = 0. Now the generalized inverse matrix *G* that satisfies equation (2) is to be found. It is demanded that $\varepsilon<10^{-6}$.

With the proposed method, *G* can be got as:

$$G = \begin{pmatrix} -0.079210 & -0.084060 & -0.075883 \\ 0.103465 & 0.109800 & 0.099118 \\ 0.137841 & 0.146280 & 0.132049 \end{pmatrix}.$$

GAG is given below with a view to check it against the object matrix so as to test the result:

$$GAG = \begin{pmatrix} -0.079211 & -0.084060 & -0.075882 \\ 0.103466 & 0.109800 & 0.099118 \\ 0.137841 & 0.146280 & 0.132049 \end{pmatrix}.$$

It is shown that under given precision demand *GAG* = *G*.

C. Generalized Inverse Matrix $A^{(3)}$ that Satisfies Equation (3) $(GA)^H = GA$

A single-layer neural network is adopted. Input matrix is *G*, weight matrix is *A*, and regard the transposition matrix of output matrix of the network (Only situation of $A\epsilon R^{m\times n}$ is discussed here) as object matrix. Train all elements' value of *G* to make network output match the object. Here,

$$E = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\sum_{k=1}^{m} g_{jk}\, a_{kj} - \sum_{k=1}^{m} g_{jk}\, a_{ki}\right)^2 \qquad (9)$$

It is derived that

$$\frac{\partial E}{\partial g_{ij}} = 2\sum_{p=1}^{n}\left\{a_{jp}\left(\sum_{k=1}^{m} g_{jk}\, a_{kp} - \sum_{k=1}^{m} g_{pk}\, a_{ki}\right)\right\} \qquad (10)$$

D. Generalized Inverse Matrix $A^{(4)}$ that Satisfies Equation (4) $(AG)^H = AG$

The method is similar to Section C. Here,

$$E = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\left(\sum_{k=1}^{n} a_{ik}\, g_{kj} - \sum_{k=1}^{n} a_{jk}\, g_{ki}\right)^2 \qquad (11)$$

And it is derived that

$$\frac{\partial E}{\partial g_{ij}} = 2\sum_{p=1}^{m}\left\{a_{pi}\left(\sum_{k=1}^{n} a_{pk}\, a_{kj} - \sum_{k=1}^{n} a_{jk}\, g_{kp}\right)\right\} \qquad (12)$$

E. Generalized Inverse Matrix $A^{(3,4)}$ that Satisfies Equation (3) and (4)

In simulation test with computer program, the above two processes of Section C and D are successively fulfilled. When two errors *E* are both

International Journal of Information Technology and Business Management
29th December 2016. Vol.55 No.1

ISSN 2304-0777 www.jitbm.com

less than the precision demand ε, the matching ends; otherwise repeat fulfilling these two procedures.

E3. $$A = \begin{pmatrix} 1.000000 & 3.000000 \\ 2.000000 & 2.000000 \\ 4.000000 & 4.000000 \\ 2.000000 & 0.000000 \end{pmatrix}.$$

It is a rectangle matrix. The generalized inverse matrix G that satisfies equation (3) and (4) is to be found. It is demanded that $\varepsilon<10^{-6}$.

With the proposed method, G can be got as:

$$G = \begin{pmatrix} 0.250000 & 0.088953 & 0.177905 & -0.116573 \\ -0.018479 & 0.124696 & 0.249393 & 0.205525 \end{pmatrix}$$

And $(AG)^H$, AG, $(GA)^H$, GA are given as the test evidence:

$$(AG)^H = \begin{pmatrix} 0.194562 & 0.463041 & 0.926083 & 0.500000 \\ 0.463040 & 0.427297 & 0.854595 & 0.177906 \\ 0.926082 & 0.854594 & 1.709189 & 0.355809 \\ 0.500002 & 0.177904 & 0.355809 & -0.233145 \end{pmatrix},$$

$$AG = \begin{pmatrix} 0.194562 & 0.463040 & 0.926082 & 0.500002 \\ 0.463041 & 0.427297 & 0.854594 & 0.177904 \\ 0.926083 & 0.854595 & 1.709189 & 0.355809 \\ 0.500000 & 0.177906 & 0.355809 & -0.233145 \end{pmatrix},$$

$$(GA)^H = \begin{pmatrix} 0.906379 & 1.639532 \\ 1.639524 & 1.191524 \end{pmatrix},$$

$$GA = \begin{pmatrix} 0.906379 & 1.639524 \\ 1.639532 & 1.191524 \end{pmatrix}.$$

It is shown that under given precision $(AG)^H = AG$ and $(GA)^H = GA$.

3.3 Discussions

There is still another definition of inverse matrix, which is left inverse and right inverse: Given $A \in C^{m\times n}$, if there exists matrix $A_L^{-1} \in C^{n\times m}$ (or $A_R^{-1} \in C^{n\times m}$) that satisfies $A_L^{-1}A = E_n$ (or $AA_R^{-1} = E_m$), A_L^{-1} (or A_R^{-1}) is called for the *Left Inverse* (or *Right Inverse*) matrix of A. It is shown that when $m = n$, if there exists the inverse of A, the left or the right inverse of A is right common inverse matrix A^{-1}.

Similarly, the corresponding SCNN structure is designed to find left inverse and right inverse, too, and are tested by computer program that proves the method to be correct.

It should be pointed out that since the above 5 kinds of generalized inverse matrices are all existing but not solitary when det$A$ = 0 or A is a rectangle matrix, what calculation result gives is just one of that kind of generalized inverse matrix. If change initial value or convergence step η, other generalized inverse matrices that satisfies that definition can be got.

3.4 Advantages of this Approach

Advantages of the approach presented in this paper are as follows:

(i) By virtue of forward optimization, generalized inverse matrices of matrices including abnormal square matrices or rectangle ones can be worked out.

(ii) Similar network structure is employed while finding the solution of different kinds of generalized inverse, and the method of network control is simple, or to say, the SCNN model has satisfactory generality.

(iii) Enough precision can be reached, and the convergence speed is fast.

(iv) Since batch-processing is used, all pairs of input-output patterns learn to adjust the weights at the same time, thus there is no "phenomenon of forgetting" between different pairs of patterns.

(v) Owing to parallel calculation, and the connection mode of all neurons is fixed, thus there is the probability of parallel processing with multi-CPU; if this approach could be realized by hardware, the speed of calculation of high-ranked generalized inverse matrix would be greatly improved. This awaits further research of technology.

To sum up, the method for finding inverse by SCNN has such advantages that it is especially suitable to be realized by computer, so it might be popularized to solve most matrix algebra problems.

4 CONCLUSIONS

Solving problem of matrix calculation with SCNN has advantages such as generality, forward calculation, parallel structure and therefore suitable to be realized by hardware, etc.

ISSN 2304-0777 www.jitbm.com

This paper has given a deep and careful research to problem of finding inverse of a matrix, the definition of generalized inverse matrix is introduced, and on the basis of SCNN and BP learning algorithm, a two-layer neural network model that can solve many kinds of generalized inverse matrices has been designed, and the corresponding concrete learning algorithm has been derived. The simulation test with computer program is proved successful. This method can be generally applied, and the computation precision is enough.

Finding the solution of 5 kinds of generalized inverse matrices $A^{(1)}$, $A^{(2)}$, $A^{(3)}$, $A^{(4)}$ and $A^{(3,4)}$, with computer is proved successful, and generalized inverse of matrices including abnormal square matrices and rectangle ones can be found. The results given in this paper show that the network structure and the calculation method designed is correct.

References

[1] Bao, Y.B., Wang, Q.W., Liu, B.F. (2015). The Weighted Moore-Penrose Inverse of a Matrix over Antirings. Proceedings of the 3rd International Workshop on Matrix Analysis and Applications, 8-11.

[2] Bersini, H., Saerens, M., Sotelino, L.G. (1994). Hopfield Net Generation, Encoding and Classification of Temporal Trajectories. IEEE Transactions on Neural Networks, 5(6), 945-953.

[3] Chen, S.C., Dai, Q. (2005). Discounted Least Squares-Improved Circular Back-Propagation Neural Networks with Applications in Time Series Prediction. Neural Computing & Applications, 14(3), 250-255.

[4] Cocuzza, S., Pretto, I., Debei, S. (2016). Least-Squares-Based Reaction Control of Space Manipulators. Journal of Guidance, Control, and Dynamics, 35(3), 976-986.

[5] Huang, C.L., Chen, Y.H., John Wan, T.L. (2012). Optimization of Data Mining in Dynamic Environments based on a Component Search Neural Network Algorithm. Journal of Convergence Information Technology, 7(7), 216-223.

[6] Huynh, H.T. and Won, Y. (2008). Weighted Least Squares Scheme for Reducing Effects of Outliers in Regression based on Extreme Learning Machine. JDCTA: International Journal of Digital Content Technology and its Applications, 2(3), 40-46.

[7] Jiao, L.C. (2003). Calculation of Neural Network, 1st edition. Xi'an: Press of Xi'an University of Electronic Science and Technology.

[8] Li, Z.L. (1999). Matrix Theory and Algebra Groundwork, 1st Edition. Chengdu: Press of Chengdu University of Electronic Science and Technology.

[9] Liu, G.H., Teng, C.L., Dong, B.B., Chen, L.L., Jiang, Y. (2010). Robust Control of Induction Motor Speed Regulation System Based on Fuzzy Neural Network Generalized Inverse. Proceedings of the 22nd Chinese Control and Decision Conference, 2729-2732.

[10] Mead, C. (1989). Analog VLSI and Neural Systems. Addison-Wesley, USA.

[11] Milighetti, G., Vallone, L., Luca, A.D. (2011). Adaptive Predictive Gaze Control of a Redundant Humanoid Robot Head. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 3192-3198.

[12] Potthast, R., Graben, P.B. (2009). Inverse Problems in Neural Field Theory. SIAM Journal on Applied Dynamical Systems, 8(4), 1405-1433.

[13] Wang, G.R., Wei, Y.M, Qiao, S.Z. (2006). Generalized Inverses: Theory and Computations. Science Press, China.

[14] Wei, Y.M. (2000). Recurrent Neural Networks for Computing Weighted Moore-Penrose Inverse. Applied Mathematics and Computation, 116(3), 279-287.

[15] Wu, X.D., Wu, H., Han, G.Q., An, Y.S. (2016). Predicting the Solubility of Sulfur in Hydrogen Sulfide Using a Back-Propagation Neural Network. International Journal of Advancements in Computing Technology, 4(8), 281-287.

[16] Xiong, Z.P., Zheng, B. (2011). The Reverse Order Laws and the Mixed-Type Reverse Order Laws for Generalized Inverses of Multiple Matrix Products. Electronic Journal of Linear Algebra, 22, 1085-1105.

[17] Zhang, Y.N., Guo, X.J., Ma, W.M. (2008). Modeling and simulation of Zhang neural network for online linear time-varying equations solving based on Matlab Simulink. Proceedings of the 7th International Conference on Machine Learning and Cybernetics, 805-810.

[18] Zhao, X.M. (2015). An Artificial Neural Network and Parallel Learning Algorithm for Matrix Calculation and Finding Solution of Matrix Equations. Proc. of the 1st Conference of China on Neural Network.