Korean J. Chem. Eng., 34(1), 170-178 (2017)
DOI: 10.1007/s11814-016-0245-9

pISSN: 0256-1115
eISSN: 1975-7220

Modeling of the phase equilibria of aqueous two-phase systems using three-dimensional neural network

Hui Chao Lv† and Da Yong Tian

School of Chemical & Environmental Engineering, Anyang Institute of Technology, China
(*Received 3 April 2016 • accepted 29 August 2016*)

Abstract−A three-dimensional neural network model has been designed for representing the phase equilibrium data related to aqueous two-phase systems. The polyvinyl pyrrolidone/phosphate/water system was selected as the model system to demonstrate the point of interest. The collected experimental data were categorized into two subsets, training and validation sets, not only to find the suitable network configuration but also to prevent the overfitting problem. Meanwhile, the weight comparison method was proposed to optimize the three-dimensional neural net. The results of accuracy comparison indicate that it outperforms the two-dimensional neural network on some details and can further enhance the calculation accuracy of the phase equilibrium data for these investigated aqueous two-phase systems. The development of the neural network in the three-dimensional space should be a research project of concern.

Keywords: Three-dimensional Neural Network, Weight Comparison Method, Aqueous Two-Phase System, Phase Equilibria

INTRODUCTION

Aqueous two-phase system (ATPS), which consists of two immiscible aqueous phases containing either two incompatible water-soluble polymers (dextran and polyethylene glycol, PEG) or one polymer and one salt (PEG and sodium phosphate), has proven to be an effective and environmental-friendly medium for the separation and purification of bioactive materials [1-3], recovery of bionanoparticles, and extraction of metal ions [4,5]. In recent years, the measurement, correlation and prediction of the phase equilibrium data of ATPS have always attracted the interest of researchers. Since direct experimental measurements of phase equilibrium data are often expensive, time-consuming and sometimes only a very limited amount of data is available, accurate description and prediction of the phase equilibrium data of ATPS is helpful to assist the designing of separation apparatus and optimization of related industrial process.

As far as a theoretical model is concerned, the osmotic virial equation, derived from the knowledge of the osmotic pressure of solvent in solution, has been employed to describe the phase behavior of ATPS [6-8]. King et al. presented a molecular thermodynamic model based on the osmotic virial equation to investigate the phase behavior for dilute aqueous mixtures containing polymers and protein [7]. Zafarani-Moattar and Sadeghi used the osmotic virial equation to correlate the phase equilibrium data of the PEG-sodium dihydrogen phosphate and PEG-disodium hydrogen phosphate ATPSs [8]. Another type of model that can be utilized to represent the phase equilibrium data of ATPS such as NRTL and UNIQUAC equations is based on the Flory-Huggins theory, which assumes the solution consists of many lattice sites occupied by solvent or solute molecules [9-12]. For example, Castro and Aznar described the phase behavior of the PEG-magnesium sulfate and PEG-sodium sulfate systems by means of the NRTL model [9]. Gautam and Simon utilized the lattice model to simulate the phase equilibrium data of the ATPS formed by PEG and potassium phosphate [10]. For these types of models, a series of adjustable parameters denoting the interactions between two components in mixture are necessary to obtain acceptable accuracy. These parameters must be determined by fitting the model to experimental data, making these models excessively dependent on experiment. Moreover, a set of specific parameters are exclusively suitable for a particular system but not valid for others. For example, the model with the parameters determined from PEG-sodium dihydrogen phosphate ATPS can yield a good agreement with experimental data for this system, but it leads to a poor result for PEG-disodium hydrogen phosphate system with the same parameters. To circumvent these drawbacks, a simpler effective method is needed.

In the past decade, due to the peculiar operation mode and inherent ability to capture the complicated relations among the variables of the problem handled, artificial neural network (ANN) has gained extensive applications in the chemical or biochemical field, such as estimation of solid solubilities in supercritical carbon dioxide [13], assisting the drug design [14], calculation of the surface tensions of binary hydrocarbon mixtures [15], prediction of the density of pure hydrocarbons [16], simulation of extracting natural dyes from plant seeds [17], predicting the advanced oxidation of organics in water matrix [18], modeling the oil bubble point pressure as well as the rejection of neutral and ionic organic compounds [19,20]. It can accurately predict the target data on the basis of adequate training. Furthermore, no involvement of professional theory makes it easier to understand than some traditional thermodynamic models. However, ANN is not flawless. Related works

†To whom correspondence should be addressed.
E-mail: prolvhuichao@126.com

have revealed that the parameters of ANN are not easy to interpret due to its unsubstantial theoretical basis [21,22]. On the other hand, ANN is an inspired model emulating the functions of the human brain that consists of many neurons assembled in a three-dimension mode. However, most of the current progress on the research work with respect to ANN is confined to two-dimensional form, as far as we know. In this regard, the research on ANN should take an important stride towards the three-dimensional space.

This article makes an attempt to conceive and design a three-dimensional neural network (TDNN) model to represent the liquid-liquid equilibria of the ATPSs formed by polyvinyl pyrrolidone (PVP, another hydrophilic polymer) and several phosphates. First, the appropriate input and output variables were selected based on the phase rule and other practical needs. Then, the experimental data were divided into two subsets to train and validate the neural network for the purpose of obtaining the suitable number of neurons in the hidden layer and preventing the overfitting problem, respectively. After that, the TDNN model was established to achieve satisfactory calculation accuracy. Meanwhile, reasonable interpretations were given for the model parameters according to the specific situations. The validation agreement plots of the output variables as well as the comparison between the calculated and experimental data for tie lines and binodal curves indicate the ability of the proposed model to describe the phase behavior of these ATPSs. The experimental data used for training and validation of the model were reported by Zafarani-Moattar and Sadeghi for PVP_{10000}-NaH_2PO_4, PVP_{10000}-Na_2HPO_4 and PVP_{10000}-Na_3PO_4 ATPSs (10000 denotes the molecular weight of PVP) [23,24]. These PVP-salt ATPSs, as mentioned by these authors, could be the alternatives to the conventional PEG-salt ATPSs, especially for the separation and purification of large biomolecules, such as proteins and enzymes, in the downstream processing of biomolecules.

1. Two-dimensional Neural Network

1-1. Outline of Artificial Neural Network

A basic framework of ANN, shown in Fig. 1, consists of some neuron nodes in different layers, namely input, hidden and output layers. The input layer receives the required data (e_i) from the real world, the hidden layer uses its neurons to process these data and then sends them (z_j) to the output layer, and the output layer writes the outputs (o_k) as dependent variables. Besides the neuron, these tasks are fulfilled by means of other elements of the neural network such as the weight connection (w_{ji}, w_{kj}), bias (b_{hj}, b_{ok}) and transfer function ($f_h(x)$, $f_o(x)$). A typical training pattern of the neural network that can teach the model about the input-output relationships generally comprises two parts: first, the inputs are delivered from the input layer to the output layer. Some related mathematical expressions are:

$$z_j = f_h\left(\sum_i w_{ji} \cdot e_i + b_{hj}\right) \tag{1}$$

$$o_k = f_o\left(\sum_i w_{kj} \cdot z_j + b_{ok}\right) \tag{2}$$

$$f_h(x) = 1/(1+1/\exp(x)) \tag{3}$$

$$f_o(x) = x \tag{4}$$

Second, the errors between target outputs and calculated ones are obtained and propagated backwards in order to adjust the parameters (weights and biases) until the neural network can yield acceptable accuracy. The two parts are therefore summarized as feed forward and back propagation, respectively.

1-2. Selection of the Input and Output Variables

Both the input and output variables are sourced from the simulated object. The proper arrangement for them is helpful to describe the phase behavior of the ATPSs. In the present study, several relevant factors are given overall consideration:

1) The selected inputs must have a real relation to the outputs. As we know, the ionic species in aqueous solution are usually "wrapped" in water molecules and this phenomenon is described as the hydration of ion. In the polymer-inorganic salt aqueous two-phase system, the stronger the hydration of inorganic salt, the stronger effect of the salting out on the polymer (such as PEG or PVP) leading to phase separation [24]. The extent of the hydration is closely related to the ionic radius and valency of ion:

(a) For the ions with the same valency such as Na^+, K^+ and NH_4^+, Na^+ has larger charge density and coulombic interaction with water molecules due to its smaller ionic radius (Na^+, K^+ and NH_4^+ have the radii of 0.098, 0.133 and 0.137 nm, respectively) [25]. As a result, Na^+ can hydrate more water molecules, and the hydrated radius of Na^+ exceeds those of K^+ and NH_4^+. For the inorganic salts containing same anion, Na^+ salt is more effective than K^+ and NH_4^+ salts in salting out the polymer to form ATPS [25].

(b) For the ions with different valencies, the valency rather than ionic radius plays a more important role in forming ATPS. Some related findings reveal that the higher valency of anion implies the stronger hydration and lower concentration required to form the phase separation [24,26]. For example, Na_3PO_4 outperforms Na_2SO_4 and NaOH in the tendency of the anions to form ATPS [26].

Since the capacity of inorganic salt to form ATPS is related to the ionic radius and ion valency, these parameters may be suitable for the input variables of a neural network, which can teach the model about the effect of salt on the phase separation. For the phosphates investigated herein, they share a common cation (Na^+)

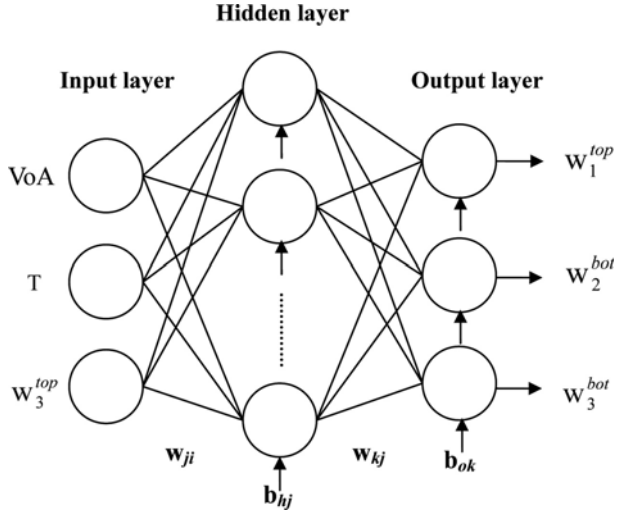

Fig. 1. A basic framework of two-dimensional neural network.

but contain different anions (PO_4^{3-}, HPO_4^{2-} and $H_2PO_4^-$) with different valencies, which conforms to the second case stated above. Therefore, we select the valency of anion (VoA) as an input variable of the neural network. On the other hand, the change of the temperature can also affect the phase behavior and equilibrium composition of these systems.

2) The phase rule reveals the freedom that represents the number of the independent variables of the system. It is indicated from the phase rule that one variable of (w_1^{top}, w_2^{top}, w_3^{top}, the weight fractions of three components in top phase) and (w_1^{bot}, w_2^{bot}, w_3^{bot}, those in bottom phase) is independent for a particular ATPS at a fixed temperature [27]. w_1 and w_2 (top or bot), the weight fractions of PVP and salt in the ATPSs, are usually employed in the rectangular phase diagram to demonstrate the model precision by comparison of the calculated and experimental data. As a result, these variables are not suitable to act as the inputs that cannot produce the calculated values. Moreover, the concentrations of salt in top phase and PVP in bottom phase (w_2^{top}, w_1^{bot}) are very small. The direct calculation utilizing the ANN model is not an appropriate strategy to obtain satisfactory accuracy according to our previous work [28]. In this work, w_2^{top} and w_1^{bot} were obtained from the normalized equations:

$$w_2^{top}=1-w_1^{top}-w_3^{top} \quad (5)$$

$$w_1^{bot}=1-w_2^{bot}-w_3^{bot} \quad (6)$$

From these instructions and accuracy requirement, we selected VoA in salt, temperature (T) and the weight fraction of the water in top phase (w_3^{top}) as the input (independent) variables. The remaining (w_1^{top}, w_2^{bot}, w_3^{bot}) were output (dependent) variables (Fig. 1). Accordingly, the node numbers of the input and output layers in the neural network were defined by the numbers of independent and dependent variables, respectively. Note that VoA and T need to be proportionally scaled to the interval (0, 1) for the convergence of the neural network (Table 1).

1-3. Hidden Layer and System Error

Although the hidden layer has no direct access to the real world, it plays a vital role in processing and conveying the input data, as well as ensuring the convergence of the network. Moreover, a suitable system error is also necessary and can be obtained from the network training. In this work, eleven sets of data were divided into two groups, which resulted in eight and three data sets for training and validation, respectively. The latter was used to prevent the network from overfitting and assess its predicting capability. More details are listed in Table 1. The algorithm tried to minimize the system error for achieving high precision. It was finally identified as 0.00105. Here the system error is given by (N_P is the number of the training samples):

$$E=\frac{1}{2}\sum_{I=1}^{N_P}\left[(w_{1,\,cal}^{top}-w_{1,\,exp}^{top})_I^2+\sum_{K=2}^{3}(w_{K,\,cal}^{bot}-w_{K,\,exp}^{bot})_I^2\right] \quad (7)$$

After that, the analysis of the node number of the hidden layer revealed that if it is less than 5, the network failed to converge because of simplistic topology structure. Conversely, the network can converge and the change of the node number of the hidden layer did not significantly affect, except the convergence rate, the total standard deviation (TSD) which stabilized at around 5.400×10^{-3} (Table 2, the letter "m" denotes the node number of the hidden layer). Here the TSD involving both the training and validation data sets can be calculated from the following expression (N_T is the number of the total samples):

Table 1. Training and validation data sets used in this work

	System	T/K	Proportion	Scaled value	VoA	Proportion	Scaled value
Training data sets	$PVP_{10000}+NaH_2PO_4+H_2O$	293.2	1000	0.2932	1	10	0.1
	$PVP_{10000}+NaH_2PO_4+H_2O$	298.2	1000	0.2982	1	10	0.1
	$PVP_{10000}+Na_2HPO_4+H_2O$	298.2	1000	0.2982	2	10	0.2
	$PVP_{10000}+Na_2HPO_4+H_2O$	318.2	1000	0.3182	2	10	0.2
	$PVP_{10000}+Na_2HPO_4+H_2O$	328.2	1000	0.3282	2	10	0.2
	$PVP_{10000}+Na_3PO_4+H_2O$	308.2	1000	0.3082	3	10	0.3
	$PVP_{10000}+Na_3PO_4+H_2O$	318.2	1000	0.3182	3	10	0.3
	$PVP_{10000}+Na_3PO_4+H_2O$	328.2	1000	0.3282	3	10	0.3
Validation data sets	$PVP_{10000}+NaH_2PO_4+H_2O$	303.2	1000	0.3032	1	10	0.1
	$PVP_{10000}+Na_2HPO_4+H_2O$	308.2	1000	0.3082	2	10	0.2
	$PVP_{10000}+Na_3PO_4+H_2O$	298.2	1000	0.2982	3	10	0.3

Table 2. Comparison of the total standard deviations between the two-dimensional and three-dimensional neural networks

M	5	6	7	8	9	10
$TSD\times10^3$ (two-dimensional)	5.426	5.440	5.617	5.409	5.366	5.376
$TSD\times10^3$ (three-dimensional)	/	5.304	5.453	5.390	4.870	4.391
M	11	12	13	14	15	16
$TSD\times10^3$ (two-dimensional)	5.125	5.448	5.375	5.414	5.454	5.130
$TSD\times10^3$ (three-dimensional)	4.348	3.474	3.436	3.478	3.581	3.705

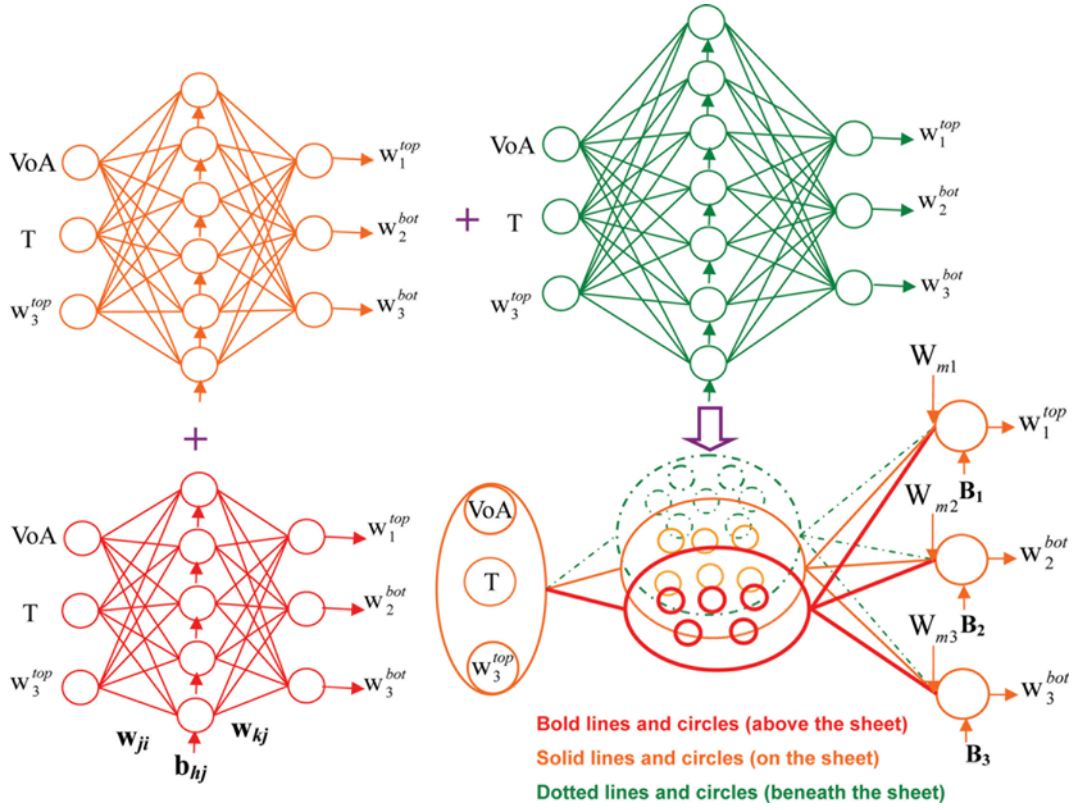

Fig. 2. The schematic of three-dimensional neural network.

$$TSD=\left\{\frac{\sum_{I=1}^{N_T}\left[\sum_{J=1}^{2}(W_{J,cal}^{top}-w_{J,exp}^{top})_I^2+\sum_{K=1}^{3}(w_{K,cal}^{bot}-w_{K,exp}^{bot})_I^2\right]}{5N_T}\right\}^{0.5} \quad (8)$$

2. Three-dimensional Neural Network

To further improve the precision of calculation, several two-dimensional neural networks were combined to form a TDNN, the input and output variables of which are the same as those of each branch network. Fig. 2, for example, displays such a tridimensional neural network formed by three branch networks (the node numbers of the hidden layer are 5, 6 and 7, respectively). Its operation process is still based on the mode employed in each two-dimensional neural network, namely, feedforward back-propagation algorithm as stated above. Note that the biases of the output layer (b_{ok}), which act as the parameters of TDNN (B) here, do not belong to each branch network. Therefore, for the calculation in each branch network, formula (2) should be correspondingly modified as:

$$o_k=f_o\left(\sum_j w_{kj}\cdot z_j\right) \quad (9)$$

Then, the outputs of three branch networks are further integrated and optimized towards the target data. The suitable parameters of the TDNN model (W and B) can be obtained from the multiple linear regression function in ORIGIN® (version 8.5):

$$w_1^{top}=\sum_{m=5}^{7}W_{m1}\cdot o_1(m)+B_1 \quad (10)$$

$$w_2^{bot}=\sum_{m=5}^{7}W_{m2}\cdot o_2(m)+B_2 \quad (11)$$

$$w_3^{bot}=\sum_{m=5}^{7}W_{m3}\cdot o_3(m)+B_3 \quad (12)$$

Summarily, the operation process of TDNN comprises two sections: first, each branch network parallelly works almost in the traditional way except for the modification in formula (2) and yields their respective outputs. After that, these outputs are assembled and optimized by the TDNN model for the further improvement of calculating precision.

RESULTS AND DISCUSSION

The suggested TDNN concept was employed to represent the phase equilibrium data of the PVP-phosphate ATPSs. The results are summarized in Table 2. With respect to TDNN, here the letter "m" denotes that TDNN contains several branch networks, the numbers of the hidden layer neurons of which range from 5 to m. For example, m=8 denotes that TDNN consists of four branch networks, the numbers of the hidden layer neurons of which are 5, 6, 7 and 8, respectively. From Table 2 the TSD presents downtrend basically before m=12, and then shows not a significant decline but a slight increase. Therefore, eight branch networks (the node numbers of the hidden layer are within the range of 5 to 12) preliminarily form the TDNN model that can yield a higher precision (TSD=3.474×10^{-3}) than ordinary two-dimensional neural network. The corresponding model parameters are listed in Table

Table 3. Parameters of the optimized three-dimensional neural network (eight branches)

M	5	6	7	8	9	10	11	12	B×10^5
W_{m1} (w_1^{top})	−1.1417	−0.7282	−7.3346	−0.1094	−1.7125	−5.6886	9.9530	7.7618	2.0234
W_{m2} (w_2^{bot})	1.5386	1.1852	5.3391	6.6672	−6.9477	−4.6845	0.8883	−2.9859	−4.0365
W_{m3} (w_3^{bot})	0.8946	−3.6419	−6.1982	6.7902	−5.1187	8.5237	13.798	−14.047	−37.189

3. In view of the different weights for a specific output variable, it is obvious that the branch networks with different node numbers of the hidden layer exert discrepant contributions on this variable. Thus, the parameters of TDNN can be interpreted as that, on one hand, each weight (value) denotes the contribution of the branch network which this weight serves to the corresponding output variable, while on the other hand, each bias (value) represents the joint effect of all the branch networks on the corresponding output variable. The integration and optimization of these branch networks can facilitate the better approximation to all the target variables. These interpretations, though plausible, could not express physicochemical meanings in theory. In this work, another TDNN model (named TDNN1) was also studied. As illustrated in Fig. 3, the salt type and temperature, two factors that affect the phase behavior of the ATPSs investigated herein, were arranged in two branch networks, respectively. Table 4 lists the corresponding model parameters. Here the weights (values) of the TDNN1 model could be theoretically interpreted as the influence coefficients of the salt type and temperature on the phase behavior of the ATPSs. From Table 4 we can also see that the phase compositions of the ATPSs are more sensitive to the change in the salt type than that in temperature, which accords with the results of some related researches [6, 24]. Unfortunately, the TDNN1 model was not adopted and further investigated owing to its low accuracy.

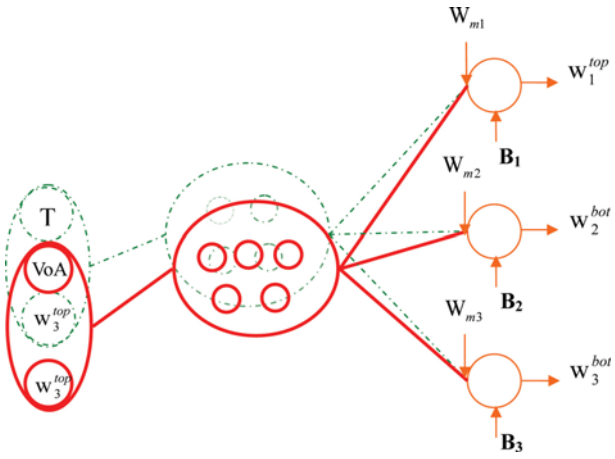

Fig. 3. The structure of the three-dimensional neural network (TDNN1, two branches).

Table 4. Parameters of the optimized three-dimensional neural network (TDNN1, two branches)

M	4 (Inputs: T, w_3^{top})	5 (Inputs: VoA, w_3^{top})	B×10^3
W_{m1} (w_1^{top})	−0.0852	1.08411	0.4226
W_{m2} (w_2^{bot})	0.31897	0.6993	2.6100
W_{m3} (w_3^{bot})	0.0839	0.9295	10.920

The structure of the TDNN model was then further simplified on condition that the TSD maintains the relative stabilization. Given that the deviations of the calculated bottom phase compositions from the experimental data are greater than those of the top phase compositions, more attention was paid to the two output variables of bottom phase compositions (w_2^{bot}, w_3^{bot}) in order to keep the TSD from being strongly affected. Here the following method was adopted: half of the eight branch networks are cut due to their relatively weak contributions to w_2^{bot} and w_3^{bot} (Table 3). More specifically, the branches (m=5, 6, 11 and 12) are cut and the branches (m=7, 8, 9 and 10) are retained for w_2^{bot}. Similarly, the branches (m=8, 10, 11 and 12) are retained for w_3^{bot}. Thus, the new model only contains six branch networks (the node numbers of the hidden layer are within the range of 7 to 12). Table 5 lists the new optimized model parameters. After several rounds of similar disposals, the simplified TDNN model is obtained and the corresponding model parameters are listed in Table 6. It consists of four branch networks and its TSD is 3.628×10^{-3}. This method simplifies the TDNN model by comparing the weights of different branch networks, so it is described as the weight comparison method.

To further show the reliability of the model, the validation agreement plot of the calculated versus the experimental outputs and the validation agreement vector [α, β, R^2] were applied to evaluate its accuracy [29]. Here, α, β, and R^2 denote the slope, intercept and correlation coefficient of the linear regression, respectively. The ideal validation agreement vector value is [1, 0, 1]. From Fig. 4, 5, and 6 it can be seen that the simplified TDNN model can provide a satisfactory description of the experimental data with the validation agreement vector equals to [1.0055, −0.0017, 0.9993], [0.9969, −0.00003, 0.9974] and [0.9975, 0.0022, 0.9975] for w_1^{top}, w_2^{bot} and w_3^{bot}, respectively. Moreover, the accuracy comparison between two-dimensional neural network and TDNN was performed

Table 5. Parameters of the optimized three-dimensional neural network (six branches)

M	7	8	9	10	11	12	B×10^5
W_{m1} (w_1^{top})	−6.0076	0.5980	−4.1752	−7.4509	9.5653	8.4705	−2.5514
W_{m2} (w_2^{bot})	3.7733	0.3612	−1.7408	3.3240	−4.8500	0.1323	0.7758
W_{m3} (w_3^{bot})	−6.2940	8.2744	−7.4593	4.1431	15.228	−12.892	−60.220

Table 6. Parameters of the optimized three-dimensional neural network (four branches)

M	7	10	11	12	B×10^5
W_{m1} (w_1^{top})	−4.5014	−3.9547	4.6650	4.7909	7.6566
W_{m2} (w_2^{bot})	2.6166	2.7163	−3.5047	−0.8282	−0.5946
W_{m3} (w_3^{bot})	−5.7887	13.622	3.8010	−10.634	−12.429

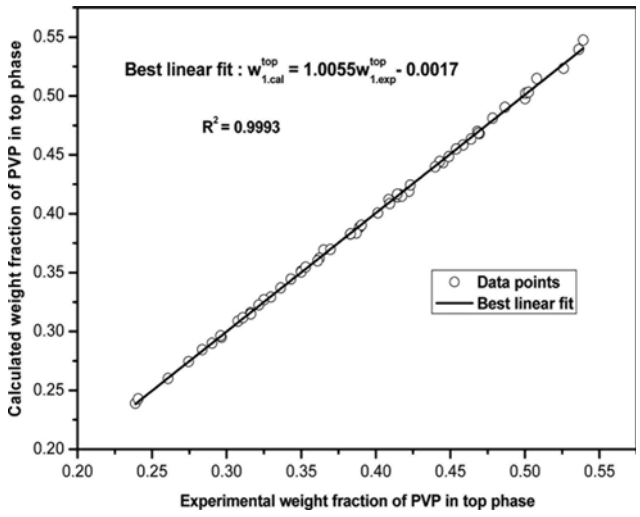

Fig. 4. Validation agreement plot of the PVP weight fraction in top phase (total data).

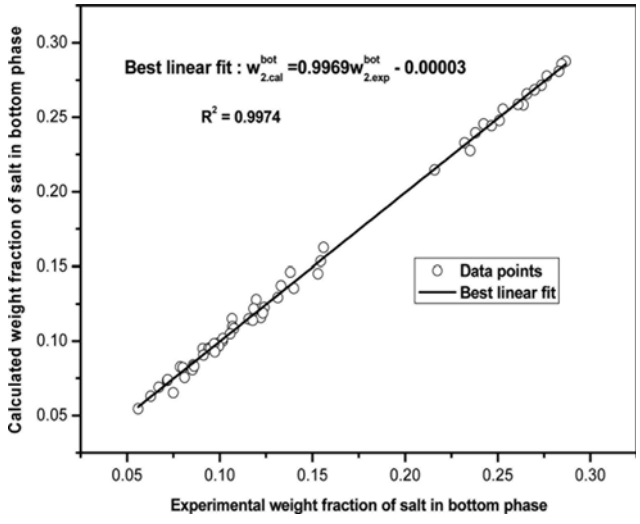

Fig. 5. Validation agreement plot of the salt weight fraction in bottom phase (total data).

for several selected systems. Table 2 shows that when the node number of the hidden layer is 11 (m=11), the two-dimensional neural network is the most precise. Therefore, {3,11,3} is determined as its topology structure for the accuracy comparison. From Fig. 7, and 8, it can be seen that for the binodal curves (phase-boundary curves) of the PVP-NaH_2PO_4 ATPS (293.2 and 303.2 K), both the two-dimensional neural network and TDNN can give satisfactory descriptions. But an important point to realize here is that the data

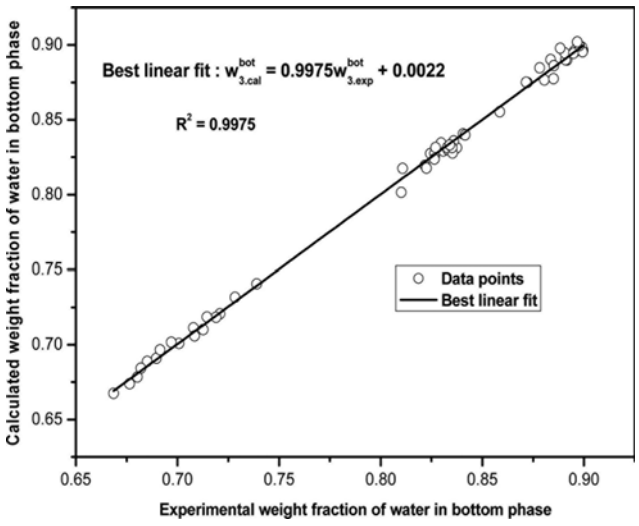

Fig. 6. Validation agreement plot of the water weight fraction in bottom phase (total data).

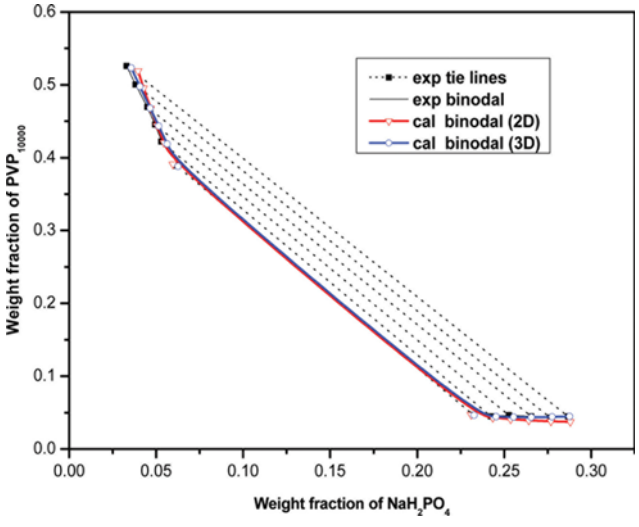

Fig. 7. Comparison of the calculated binodal curves based on the two dimensional (2D) and three-dimensional (3D) neural network models for PVP_{10000}-NaH_2PO_4-H_2O system at 293.2 K.

obtained from the latter are closer to the experimental binodal curve for each temperature. Fig. 9 presents a similar situation for the PVP-Na_2HPO_4 ATPS (308.2 K). The experimental and calculated (2D) binodal curves divaricate at the lower right corner of Fig. 9, but the calculated (3D) binodal curve is in accord with the experimental one. From Fig. 10 it can be seen that TDNN can yield better descriptions of the tie lines of the PVP-Na_3PO_4 ATPS (318.2 K). It is obvious that two shorter tie lines calculated from the two-dimensional neural network deviate from the experimental ones, which are corrected by the TDNN model (for clarity reasons, only the endpoints are shown for some calculated tie lines). On the other hand, the deviations of calculated tie lines from experimental data based on different models are summarized in Table 7. Here Dev^a, Dev^b and Dev^c denote the deviations of the two-dimensional neural network, TDNN and extended F-H+PDH thermo-

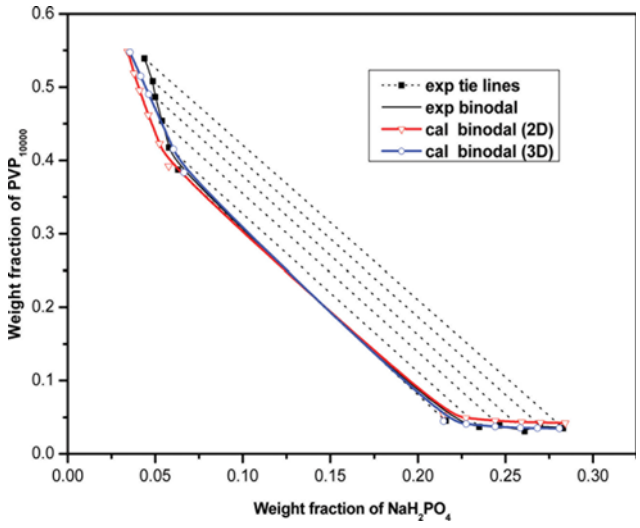

Fig. 8. Comparison of the calculated binodal curves based on the 2D and 3D neural network models for PVP_{10000}-NaH_2PO_4-H_2O system at 303.2 K.

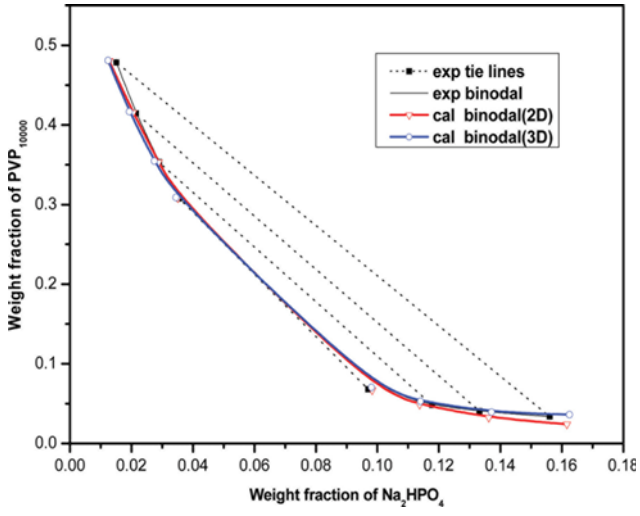

Fig. 9. Comparison of the calculated binodal curves based on the 2D and 3D neural network models for PVP_{10000}-Na_2HPO_4-H_2O system at 308.2 K.

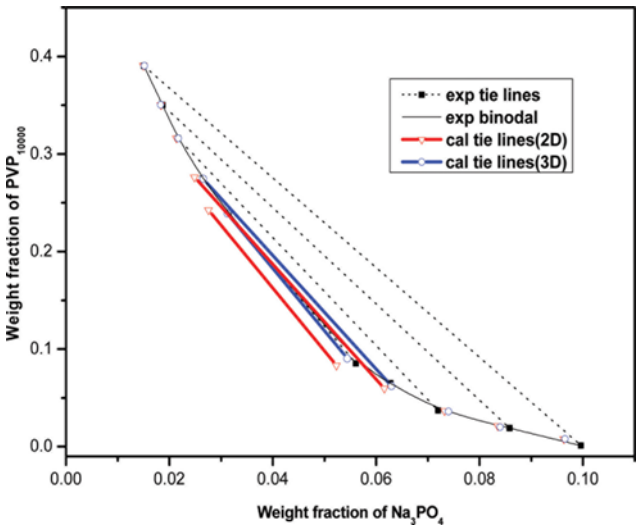

Fig. 10. Comparison of the calculated tie lines based on the 2D and 3D neural network models for PVP_{10000}-Na_3PO_4-H_2O system at 318.2 K.

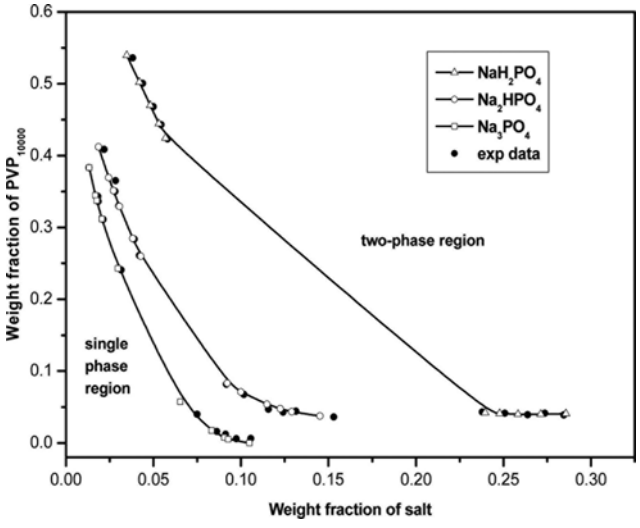

Fig. 11. The effect of the salt type on the binodal curves for the PVP_{10000}-salt-H_2O system at 298.2 K.

dynamic models (a thermodynamic equation employed in the literatures to represent the phase equilibrium data for these PVP-phosphate ATPSs [23,24]), respectively. According to the above analysis and these deviations, we concluded that the TDNN model can be more satisfactorily used to correlate and predict the phase equilibria of these ATPSs. Here the deviation for each system can be obtained from the following expression (N is the number of tie lines):

Table 7. Deviations of calculated tie-line data based on different models

System	T/K	N	$Dev^a \times 10^4$	$Dev^b \times 10^4$	$Dev^c \times 10^4$
PVP_{10000}+NaH_2PO_4+H_2O	293.2/298.2/303.2	17	0.28	0.09	0.62
PVP_{10000}+Na_2HPO_4+H_2O	298.2/308.2/318.2/328.2	20	0.26	0.17	4.95
PVP_{10000}+Na_3PO_4+H_2O	298.2/308.2/318.2/328.2	21	0.25	0.13	3.44

Dev^a, Dev^b and Dev^c are the calculated deviations by means of the two-dimensional neural network, TDNN and extended F-H+PDH equation, respectively

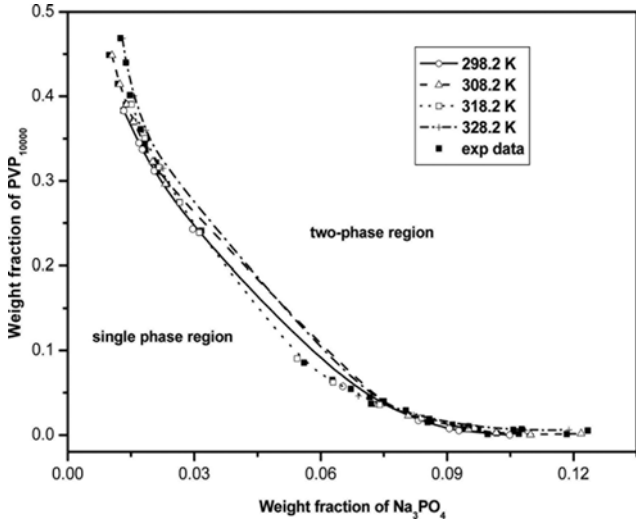

Fig. 12. The effect of temperature on the binodal curves for the PVP_{10000}-Na_3PO_4-H_2O system.

$$Dev=\left\{\frac{\sum_{I=1}^{N}\left[\sum_{J=1}^{2}(W_{J,cal}^{top}-w_{J,exp}^{top})_I^2+\sum_{K=1}^{3}(w_{K,cal}^{bot}-w_{K,exp}^{bot})_I^2\right]}{5N}\right\}^{0.5} \quad (13)$$

Finally, the effects of the salt type and temperature on the phase behavior of the ATPSs were investigated based on the proposed model. The results are illustrated in Fig. 11, and 12 in which the binodal curves are sketched via the TDNN calculated equilibrium compositions. In addition to the good agreement with the experimental data, the analysis of two figures reveals that the phase separation occurs at lower salt concentration when the system contains the salt with higher valence on the anion. This is because the higher valence anion is a better salting-out agent which hydrates more water than the lower valence anion. On the other hand, the effect of temperature on the binodal curves is not as obvious as that of the salt type. The binodal curves of PVP-Na_3PO_4 ATPS seem to be almost overlapped under four temperatures investigated here. These trends are in accord with those reported in the literature [6,24] and can be represented exactly by the proposed TDNN model. Moreover, the literature also reported that the slope of tie lines increases with the rise of temperature owing to the effect of changes in temperature on the solubility of PVP in aqueous solution [12]. Fig. 13 displays a similar situation for PVP-Na_2HPO_4 system (303.2 K, interpolated; 333.2 K, extrapolated).

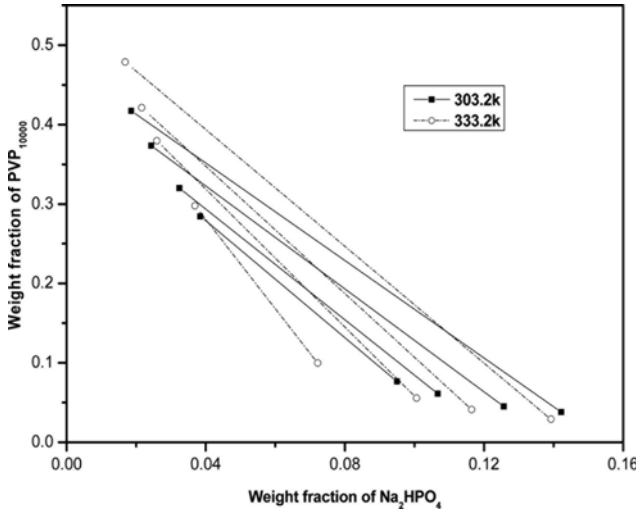

Fig. 13. The effect of temperature on the tie lines for the PVP_{10000}-Na_2HPO_4-H_2O system.

CONCLUSION

TDNN, a new version of neural network derived from the rethinking about the structure and work mechanism of the human brain as well as the limitation in the current research with regard to ANN, was designed to represent the phase equilibrium data of the ATPSs formed by PVP and several phosphates. The weight comparison method was proposed to optimize its structure. The TDNN model consists of four branch networks and can estimate the phase equilibrium data more accurately than the two-dimensional neural network and extended F-H+PDH equation. By the applications of the appropriate input and output variables, it is capable of depicting information about the phase diagram and the effects due to the changes in the salt type and temperature on the phase behavior of the ATPSs. Meanwhile, the model parameters have been given rational interpretations, although not completely physicochemical meanings.

The further research and development on the application of TDNN model in the chemical, biochemical as well as related technical field are the subject of a future study, and it might be a good strategy to deal with some intractable problems we will encounter. On the other hand, more fundamental researches on the neural network theory are required to obtain more meaningful relations between the model parameters and the simulated system, which enable a better utilization of neural network.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (No. U1404217) and a Key Science and Technology Project of Henan Province (No. 142102310190).

SUPPORTING INFORMATION

Additional information as noted in the text. This information is available via the Internet at http://www.springer.com/chemistry/journal/11814.

NOMENCLATURE

ANN : artificial neural network
ATPS : aqueous two-phase system
b_{hj} : bias of the hidden layer node
b_{ok} : bias of the output layer node
B : bias of the three-dimensional neural network
e_i : inputs of the neural network

Dev : deviation
E : system error
F-H : Flory-Huggins
m : node number of the hidden layer
N : number of tie lines
N_P : number of the training samples
N : number of the total samples
NRTL : non-random two-liquid
o_k : outputs of the neural network
PDH : Pitzer-Debye-Hückel
PEG : polyethylene glycol
PVP : polyvinyl pyrrolidone
R^2 : correlation coefficient
T : temperature [K]
TDNN : three-dimensional neural network
TSD : total standard deviation
UNIQUAC : universal quasi-chemical
VoA : valency of anion
w : weight fraction
w_{ji} : input-hidden layer connection weight
w_{kj} : hidden-output layer connection weight
W : weight of the three-dimensional neural network
z_j : calculated value of the hidden layer

Greek Letters

α : slope
β : intercept

Super/Subscripts

top : top phase
bot : bottom phase
cal : calculated
exp : experimental
1 : polyvinyl pyrrolidone (PVP)
2 : salt
3 : water

REFERENCES

1. A. M. Azevedo, P. A. Rosa, I. F. Ferreira and M. R. J. Aires-Barros, *J. Biotechnol.*, **132**, 209 (2007).
2. A. Šalić, A. Tušek, D. Fabek, I. Rukavina and B. Zelić, *Food Technol. Biotechnol.*, **49**, 495 (2011).
3. S. Tanuja, N. D. Srinivas, M. K. Gowthaman and K. S. M. S. Raghavarao, *Bioprocess. Eng.*, **23**, 63 (2000).
4. A. Negrete, T. C. Ling and A. J. Lyddiatt, *Chromatogr. B.*, **854**, 13 (2007).
5. G. D. Rodrigues, M. D. C. H. da Silva, L. H. M. da Silva, F. J. Paggioli, L. A. Minim and J. S. D. Reis Coimbra, *Sep. Purif. Technol.*, **62**, 687 (2008).
6. N. Fedicheva, L. Ninni and G. Maurer, *J. Chem. Eng. Data*, **52**, 1858 (2007).
7. R. S. King, H. W. Blanch and J. M. Prausnitz, *AIChE J.*, **34**, 1585 (1998).
8. M. T. Zafarani-Moattar and R. Sadeghi, *Fluid. Phase. Equilib.*, **181**, 95 (2001).
9. B. D. Castro and M. Aznar, *Braz. J. Chem. Eng.*, **22**, 463 (2005).
10. S. Gautam and L. Simon, *Chem. Eng. Commun.*, **194**, 117 (2007).
11. R. Sadeghi, *Fluid. Phase. Equilib.*, **237**, 40 (2005).
12. M. T. Zafarani-Moattar and V. Abdizadeh-Aliyar, *J. Chem. Thermodyn.*, **72**, 125 (2014).
13. M. Lashkarbolooki, B. Vaferi and M. R. Rahimpour, *Fluid. Phase. Equilib.*, **308**, 35 (2011).
14. Z. Zheng and K. M. Merz Jr., *J. Chem. Inf. Model.*, **51**, 1296 (2011).
15. H. Parhizgar, M. R. Dehghani, A. Khazaei and M. Dalirian, *Ind. Eng. Chem. Res.*, **51**, 2775 (2012).
16. R. Haghbakhsh, H. Adib, P. Keshavarz, M. Koolivand and S. Keshtkari, *Thermochim. Acta*, **551**, 124 (2013).
17. K. Sinha, S. Chowdhury, P. D. Saha and S. Datta, *Ind. Crop. Prod.*, **41**, 165 (2013).
18. T. Bolanča, Š. Ukić, I. Peternel, H. Kušić and A. L. Božić, *Indian J. Chem. Technol.*, **21**, 21 (2014).
19. M. Afshar, A. Gholami and M. Asoodeh, *Korean J. Chem. Eng.*, **31**, 496 (2014).
20. Y. Ammi, L. Khaouane and S. Hanini, *Korean J. Chem. Eng.*, **32**, 2300 (2015).
21. P. Kan and C. J. Lee, *Ind. Eng. Chem. Res.*, **35**, 2015 (1996).
22. M. Safamirzaei and H. Modarress, *Fluid. Phase. Equilib.*, **309**, 53 (2011).
23. M. T. Zafarani-Moattar and R. Sadeghi, *Fluid. Phase. Equilib.*, **203**, 177 (2002).
24. M. T. Zafarani-Moattar and R.Sadeghi, *Fluid. Phase. Equilib.*, **238**, 129 (2005).
25. M. T. Zafarani-Moattar and J. Gasemi, *Fluid. Phase. Equilib.*, **198**, 281 (2002).
26. K. Ananthapadmanabhan and E. D. Goddard, *Langmuir*, **3**, 25 (1987).
27. T. Qiu, P. S. Ma, L. E. Wang and H. D. Zheng, *Chem. Eng. (China)*, **32**, 62 (2004).
28. H. C. Lv and Y. H. Shen, *J. Korean Chem. Soc.*, **57**, 370 (2013).
29. C. Si-Moussa, S. Hanini, R. Derriche, M. Bouhedda and A. Bouzidi, *Braz. J. Chem. Eng.*, **25**, 183 (2008).

Supporting Information

Modeling of the phase equilibria of aqueous two-phase systems using three-dimensional neural network

Hui Chao Lv† and Da Yong Tian

School of Chemical & Environmental Engineering, Anyang Institute of Technology, China
(*Received 3 April 2016 • accepted 29 August 2016*)

This program (written in delphi 6.0), in which the numbers of the neurons in the input, hidden and output layers can be conveniently adjusted, was used to train and validate the branch neural network. Then the three dimensional neural network and optimized model parameters can be obtained based on the calculated phase equilibrium data with this program.

```
procedure TForm1.Button1Click(Sender: TObject);
const
m=3;r=12;n=3;// the numbers of the neuron in the input, hidden and output layers
datapoint=43;// the number of the training samples
datapoint1=58;// the number of the total samples
eps=0.00105;// the system error
type
tmyarray1=array[1..m,1..r] of real;//w01[1..m,1..r]
tmyarray2=array[1..r,1..n] of real;//w12[1..r,1..n]
tmyarray3=array[1..r] of real;//b1[1..r]
tmyarray4=array[1..n] of real;//b2,max,min[1..n]
tmyarray5=array[1..m,1..r,0..1] of real;//detaw01[1..m,1..r,0..1]
tmyarray6=array[1..r,1..n,0..1] of real;//detaw12[1..r,1..n,0..1]
tmyarray7=array[1..r,0..1] of real;//detab1[1..r,0..1]
tmyarray8=array[1..n,0..1] of real;//detab2[1..n,0..1]
tmyarray9=array[1..datapoint1,1..m] of real;//in0[1..datapoint1,1..m]
tmyarray10=array[1..datapoint1,1..r] of real;//in1,out1,wc1[1..datapoint1,1..r]
tmyarray11=array[1..datapoint1,1..n] of real;//in2,out2j,out2s,wc2[1..datapoint1,1..n]
var
w01:tmyarray1; w12:tmyarray2; b1:tmyarray3;
detaw01:tmyarray5;detaw12:tmyarray6;detab1:tmyarray7;
in0:tmyarray9; in1,out1,wc1:tmyarray10; in2,out2j,out2s,wc2:tmyarray11;
max,min:tmyarray4;
i,j,l,k:integer;
rms,aad:real;

begin
//PVP(10000)+NaH2PO4+H2O (293.2 K)
in0[1,1]:=0.1000;  in0[1,2]:=0.2932;  in0[1,3]:=0.5492;  out2s[1,1]:=0.3890; out2s[1,2]:=0.2322; out2s[1,3]:=0.7210;
in0[2,1]:=0.1000;  in0[2,2]:=0.2932;  in0[2,3]:=0.5248;  out2s[2,1]:=0.4222; out2s[2,2]:=0.2424; out2s[2,3]:=0.7127;
in0[3,1]:=0.1000;  in0[3,2]:=0.2932;  in0[3,3]:=0.5053;  out2s[3,1]:=0.4452; out2s[3,2]:=0.2528; out2s[3,3]:=0.7008;
in0[4,1]:=0.1000;  in0[4,2]:=0.2932;  in0[4,3]:=0.4854;  out2s[4,1]:=0.4696; out2s[4,2]:=0.2657; out2s[4,3]:=0.6896;
in0[5,1]:=0.1000;  in0[5,2]:=0.2932;  in0[5,3]:=0.4616;  out2s[5,1]:=0.5002; out2s[5,2]:=0.2765; out2s[5,3]:=0.6803;
in0[6,1]:=0.1000;  in0[6,2]:=0.2932;  in0[6,3]:=0.4410;  out2s[6,1]:=0.5260; out2s[6,2]:=0.2867; out2s[6,3]:=0.6686;
// PVP(10000)+NaH2PO4+H2O (298.2 K)
in0[7,1]:=0.1000;  in0[7,2]:=0.2982;  in0[7,3]:=0.5187;  out2s[7,1]:=0.4232; out2s[7,2]:=0.2380; out2s[7,3]:=0.7190;
in0[8,1]:=0.1000;  in0[8,2]:=0.2982;  in0[8,3]:=0.5025;  out2s[8,1]:=0.4430; out2s[8,2]:=0.2509; out2s[8,3]:=0.7079;
in0[9,1]:=0.1000;  in0[9,2]:=0.2982;  in0[9,3]:=0.4819;  out2s[9,1]:=0.4681; out2s[9,2]:=0.2639; out2s[9,3]:=0.6970;
in0[10,1]:=0.1000; in0[10,2]:=0.2982; in0[10,3]:=0.4557; out2s[10,1]:=0.5005; out2s[10,2]:=0.2736; out2s[10,3]:=0.6852;
in0[11,1]:=0.1000; in0[11,2]:=0.2982; in0[11,3]:=0.4258; out2s[11,1]:=0.5361; out2s[11,2]:=0.2845; out2s[11,3]:=0.6765;
//PVP(10000)+Na2HPO4+H2O (298.2 K)
in0[12,1]:=0.2000; in0[12,2]:=0.2982; in0[12,3]:=0.6972; out2s[12,1]:=0.2608; out2s[12,2]:=0.0917; out2s[12,3]:=0.8266;
```

in0[13,1]:=0.2000; in0[13,2]:=0.2982; in0[13,3]:=0.6772; out2s[13,1]:=0.2837; out2s[13,2]:=0.1017; out2s[13,3]:=0.8308;
in0[14,1]:=0.2000; in0[14,2]:=0.2982; in0[14,3]:=0.6402; out2s[14,1]:=0.3297; out2s[14,2]:=0.1157; out2s[14,3]:=0.8375;
in0[15,1]:=0.2000; in0[15,2]:=0.2982; in0[15,3]:=0.6219; out2s[15,1]:=0.3500; out2s[15,2]:=0.1241; out2s[15,3]:=0.8331;
in0[16,1]:=0.2000; in0[16,2]:=0.2982; in0[16,3]:=0.6065; out2s[16,1]:=0.3651; out2s[16,2]:=0.1314; out2s[16,3]:=0.8245;
in0[17,1]:=0.2000; in0[17,2]:=0.2982; in0[17,3]:=0.5695; out2s[17,1]:=0.4087; out2s[17,2]:=0.1531; out2s[17,3]:=0.8108;
// PVP(10000)+Na_2HPO_4+H_2O (318.2 K)
in0[18,1]:=0.2000; in0[18,2]:=0.3182; in0[18,3]:=0.6721; out2s[18,1]:=0.2902; out2s[18,2]:=0.0788; out2s[18,3]:=0.8265;
in0[19,1]:=0.2000; in0[19,2]:=0.3182; in0[19,3]:=0.6410; out2s[19,1]:=0.3250; out2s[19,2]:=0.0909; out2s[19,3]:=0.8298;
in0[20,1]:=0.2000; in0[20,2]:=0.3182; in0[20,3]:=0.5928; out2s[20,1]:=0.3830; out2s[20,2]:=0.1067; out2s[20,3]:=0.8360;
in0[21,1]:=0.2000; in0[21,2]:=0.3182; in0[21,3]:=0.5630; out2s[21,1]:=0.4160; out2s[21,2]:=0.1197; out2s[21,3]:=0.8270;
in0[22,1]:=0.2000; in0[22,2]:=0.3182; in0[22,3]:=0.5208; out2s[22,1]:=0.4640; out2s[22,2]:=0.1382; out2s[22,3]:=0.8218;
// PVP(10000)+Na_2HPO_4+H_2O (328.2 K)
in0[23,1]:=0.2000; in0[23,2]:=0.3282; in0[23,3]:=0.6677; out2s[23,1]:=0.2967; out2s[23,2]:=0.0812; out2s[23,3]:=0.8354;
in0[24,1]:=0.2000; in0[24,2]:=0.3282; in0[24,3]:=0.6101; out2s[24,1]:=0.3623; out2s[24,2]:=0.0991; out2s[24,3]:=0.8406;
in0[25,1]:=0.2000; in0[25,2]:=0.3282; in0[25,3]:=0.5690; out2s[25,1]:=0.4094; out2s[25,2]:=0.1223; out2s[25,3]:=0.8416;
in0[26,1]:=0.2000; in0[26,2]:=0.3282; in0[26,3]:=0.5240; out2s[26,1]:=0.4586; out2s[26,2]:=0.1400; out2s[26,3]:=0.8328;
in0[27,1]:=0.2000; in0[27,2]:=0.3282; in0[27,3]:=0.4826; out2s[27,1]:=0.5024; out2s[27,2]:=0.1546; out2s[27,3]:=0.8225;
//PVP(10000)+Na_3PO_4+H_2O (308.2 K)
in0[28,1]:=0.3000; in0[28,2]:=0.3082; in0[28,3]:=0.6805; out2s[28,1]:=0.2960; out2s[28,2]:=0.0717; out2s[28,3]:=0.8836;
in0[29,1]:=0.3000; in0[29,2]:=0.3082; in0[29,3]:=0.6574; out2s[29,1]:=0.3216; out2s[29,2]:=0.0854; out2s[29,3]:=0.8996;
in0[30,1]:=0.3000; in0[30,2]:=0.3082; in0[30,3]:=0.6147; out2s[30,1]:=0.3697; out2s[30,2]:=0.0940; out2s[30,3]:=0.8990;
in0[31,1]:=0.3000; in0[31,2]:=0.3082; in0[31,3]:=0.5959; out2s[31,1]:=0.3903; out2s[31,2]:=0.1018; out2s[31,3]:=0.8962;
in0[32,1]:=0.3000; in0[32,2]:=0.3082; in0[32,3]:=0.5734; out2s[32,1]:=0.4147; out2s[32,2]:=0.1070; out2s[32,3]:=0.8917;
in0[33,1]:=0.3000; in0[33,2]:=0.3082; in0[33,3]:=0.5412; out2s[33,1]:=0.4490; out2s[33,2]:=0.1184; out2s[33,3]:=0.8806;
// PVP(10000)+Na_3PO_4+H_2O (318.2 K)
in0[34,1]:=0.3000; in0[34,2]:=0.3182; in0[34,3]:=0.7297; out2s[34,1]:=0.2388; out2s[34,2]:=0.0561; out2s[34,3]:=0.8586;
in0[35,1]:=0.3000; in0[35,2]:=0.3182; in0[35,3]:=0.6990; out2s[35,1]:=0.2745; out2s[35,2]:=0.0628; out2s[35,3]:=0.8722;
in0[36,1]:=0.3000; in0[36,2]:=0.3182; in0[36,3]:=0.6623; out2s[36,1]:=0.3160; out2s[36,2]:=0.0720; out2s[36,3]:=0.8910;
in0[37,1]:=0.3000; in0[37,2]:=0.3182; in0[37,3]:=0.6313; out2s[37,1]:=0.3500; out2s[37,2]:=0.0858; out2s[37,3]:=0.8954;
in0[38,1]:=0.3000; in0[38,2]:=0.3182; in0[38,3]:=0.5945; out2s[38,1]:=0.3904; out2s[38,2]:=0.0996; out2s[38,3]:=0.8994;
// PVP(10000)+Na_3PO_4+H_2O (328.2 K)
in0[39,1]:=0.3000; in0[39,2]:=0.3282; in0[39,3]:=0.6625; out2s[39,1]:=0.3162; out2s[39,2]:=0.0672; out2s[39,3]:=0.8783;
in0[40,1]:=0.3000; in0[40,2]:=0.3282; in0[40,3]:=0.6216; out2s[40,1]:=0.3611; out2s[40,2]:=0.0802; out2s[40,3]:=0.8907;
in0[41,1]:=0.3000; in0[41,2]:=0.3282; in0[41,3]:=0.5838; out2s[41,1]:=0.4014; out2s[41,2]:=0.0951; out2s[41,3]:=0.8949;
in0[42,1]:=0.3000; in0[42,2]:=0.3282; in0[42,3]:=0.5464; out2s[42,1]:=0.4398; out2s[42,2]:=0.1077; out2s[42,3]:=0.8853;
in0[43,1]:=0.3000; in0[43,2]:=0.3282; in0[43,3]:=0.5186; out2s[43,1]:=0.4689; out2s[43,2]:=0.1234; out2s[43,3]:=0.8716;
//----------
///PVP(10000)+NaH_2PO_4+H_2O (303.2 K)
in0[44,1]:=0.1000; in0[44,2]:=0.3032; in0[44,3]:=0.5500; out2s[44,1]:=0.3870; out2s[44,2]:=0.2160; out2s[44,3]:=0.7390;
in0[45,1]:=0.1000; in0[45,2]:=0.3032; in0[45,3]:=0.5247; out2s[45,1]:=0.4175; out2s[45,2]:=0.2351; out2s[45,3]:=0.7283;
in0[46,1]:=0.1000; in0[46,2]:=0.3032; in0[46,3]:=0.4922; out2s[46,1]:=0.4539; out2s[46,2]:=0.2467; out2s[46,3]:=0.7146;
in0[47,1]:=0.1000; in0[47,2]:=0.3032; in0[47,3]:=0.4634; out2s[47,1]:=0.4866; out2s[47,2]:=0.2610; out2s[47,3]:=0.7085;
in0[48,1]:=0.1000; in0[48,2]:=0.3032; in0[48,3]:=0.4434; out2s[48,1]:=0.5080; out2s[48,2]:=0.2698; out2s[48,3]:=0.6915;
in0[49,1]:=0.1000; in0[49,2]:=0.3032; in0[49,3]:=0.4171; out2s[49,1]:=0.5391; out2s[49,2]:=0.2830; out2s[49,3]:=0.6821;

// PVP(10000)+Na_2HPO_4+H_2O (308.2 K)
in0[50,1]:=0.2000; in0[50,2]:=0.3082; in0[50,3]:=0.6568; out2s[50,1]:=0.3076; out2s[50,2]:=0.0970; out2s[50,3]:=0.8352;
in0[51,1]:=0.2000; in0[51,2]:=0.3082; in0[51,3]:=0.6179; out2s[51,1]:=0.3531; out2s[51,2]:=0.1178; out2s[51,3]:=0.8340;
in0[52,1]:=0.2000; in0[52,2]:=0.3082; in0[52,3]:=0.5641; out2s[52,1]:=0.4144; out2s[52,2]:=0.1332; out2s[52,3]:=0.8264;
in0[53,1]:=0.2000; in0[53,2]:=0.3082; in0[53,3]:=0.5066; out2s[53,1]:=0.4783; out2s[53,2]:=0.1561; out2s[53,3]:=0.8101;

// PVP(10000)+Na_3PO_4+H_2O (298.2 K)
in0[54,1]:=0.3000; in0[54,2]:=0.2982; in0[54,3]:=0.7275; out2s[54,1]:=0.2408; out2s[54,2]:=0.0750; out2s[54,3]:=0.8850;
in0[55,1]:=0.3000; in0[55,2]:=0.2982; in0[55,3]:=0.6678; out2s[55,1]:=0.3109; out2s[55,2]:=0.0863; out2s[55,3]:=0.8978;
in0[56,1]:=0.3000; in0[56,2]:=0.2982; in0[56,3]:=0.6453; out2s[56,1]:=0.3362; out2s[56,2]:=0.0913; out2s[56,3]:=0.8967;
in0[57,1]:=0.3000; in0[57,2]:=0.2982; in0[57,3]:=0.6385; out2s[57,1]:=0.3432; out2s[57,2]:=0.0974; out2s[57,3]:=0.8967;

```
in0[58,1]:=0.3000;   in0[58,2]:=0.2982;   in0[58,3]:=0.6037;   out2s[58,1]:=0.3831; out2s[58,2]:=0.1057; out2s[58,3]:=0.8883;



randseed:=0;
for i:=1 to m do
 for j:=1 to r do
 begin
 w01[i,j]:=random-0.5;
 detaw01[i,j,1]:=0.0;
 end;
for i:=1 to r do
 for j:=1 to n do
 begin
 w12[i,j]:=random-0.5;
 detaw12[i,j,1]:=0.0;
 end;
for i:=1 to r do
 begin
 b1[i]:=random-0.5;
 detab1[i,1]:=0.0;
 end;
for i:=1 to m do
 for j:=1 to r do
 begin
 detaw01[i,j,0]:=detaw01[i,j,1];
 detaw01[i,j,1]:=0.0;
 end;
for i:=1 to r do
 for j:=1 to n do
 begin
 detaw12[i,j,0]:=detaw12[i,j,1];
 detaw12[i,j,1]:=0.0;
 end;
for i:=1 to r do
 begin
 detab1[i,0]:=detab1[i,1];
 detab1[i,1]:=0.0;
 end;
k:=1; rms:=0.0;
repeat
for j:=1 to r do
  in1[k,j]:=0.0;
for j:=1 to r do
  for l:= 1 to m do
  in1[k,j]:=in1[k,j]+w01[l,j]*in0[k,l];
for j:=1 to r do
  in1[k,j]:=in1[k,j]+b1[j];
for j:=1 to r do
  out1[k,j]:=1.0/(1+exp(-in1[k,j]));
for j:= 1 to n do
  in2[k,j]:=0.0;
for j:= 1 to n do
 for l:= 1 to r do
   in2[k,j]:=in2[k,j]+w12[l,j]*out1[k,l];
for j:=1 to n do
```

```
 begin
 out2j[k,j]:=in2[k,j];
 rms:=rms+0.5*power((out2j[k,j]-out2s[k,j]),2);
 wc2[k,j]:=(out2j[k,j]-out2s[k,j]);
 end;
 k:=k+1;
until  k>datapoint;
if rms>eps then
begin
for i:=1 to datapoint do
 for j:= 1 to r do
 wc1[i,j]:=0.0;
for i:=1 to datapoint do
  for j:=1 to r do
   for l:= 1 to n do
   wc1[i,j]:=wc1[i,j]+w12[j,l]*wc2[i,l];
for i:=1 to datapoint do
  for j:=1 to r do
  begin
  wc1[i,j]:=wc1[i,j]*exp(-in1[i,j]);
  wc1[i,j]:=wc1[i,j]/power(1.0+exp(-in1[i,j]),2);
  end;
for i:=1 to m do
 for j:=1 to r do
  for l:=1 to datapoint do
  detaw01[i,j,1]:=detaw01[i,j,1]+in0[l,i]*wc1[l,j];
for i:=1 to m do
 for j:=1 to r do
  w01[i,j]:=w01[i,j]-0.5*detaw01[i,j,1]/datapoint-0.15*detaw01[i,j,0]/datapoint;
for i:=1 to r do
 for j:=1 to n do
  for l:=1 to datapoint do
  detaw12[i,j,1]:=detaw12[i,j,1]+out1[l,i]*wc2[l,j];
for i:=1 to r do
 for j:=1 to n do
 w12[i,j]:=w12[i,j]-0.5*detaw12[i,j,1]/datapoint-0.15*detaw12[i,j,0]/datapoint;
for i:=1 to r do
  for l:=1 to datapoint do
  detab1[i,1]:=detab1[i,1]+wc1[l,i];
for i:=1 to r do
b1[i]:=b1[i]-0.5*detab1[i,1]/datapoint-0.15*detab1[i,0]/datapoint;
end;
until rms<=eps;
edit21.text:=inttostr(k);
//---------------------------------------------------------------------------------------
repeat
for j:=1 to r do
 in1[k,j]:=0.0;
for j:=1 to r do
 for l:= 1 to m do
 in1[k,j]:=in1[k,j]+w01[l,j]*in0[k,l];
for j:=1 to r do
  in1[k,j]:=in1[k,j]+b1[j];
for j:=1 to r do
  out1[k,j]:=1.0/(1+exp(-in1[k,j]));
for j:= 1 to n do
```

```
  in2[k,j]:=0.0;
for j:= 1 to n do
 for l:= 1 to r do
   in2[k,j]:=in2[k,j]+w12[l,j]*out1[k,l];
for j:=1 to n do
  out2j[k,j]:=in2[k,j];
k:=k+1;
until  k>datapoint1;
for i:=1 to datapoint1 do
begin
memo1.lines.append(floattostr(out2j[i,1]));
memo2.lines.append(floattostr(out2s[i,1]));
memo3.lines.append(floattostr(out2j[i,2]));
memo4.lines.append(floattostr(out2s[i,2]));
memo5.lines.append(floattostr(out2j[i,3]));
memo6.lines.append(floattostr(out2s[i,3]));
memo7.lines.append(floattostr(in0[i,3]));
if i=datapoint then
begin
memo1.lines.append('Following are validation data!');
memo2.lines.append('Following are validation data!');
memo3.lines.append('Following are validation data!');
memo4.lines.append('Following are validation data!');
memo5.lines.append('Following are validation data!');
memo6.lines.append('Following are validation data!');
memo7.lines.append('Following are validation data!');
end;
end;
aad:=0.0;
for i:= 1 to datapoint do
  for j:=1 to n do
begin
aad:=aad+power((out2j[i,j]-out2s[i,j]),2);
end;
aad:=sqrt(aad/(datapoint*n));
edit19.text:=floattostr(aad);
aad:=0.0;
for i:=(datapoint+1) to datapoint1 do
   for j:=1 to n do
begin
aad:=aad+power((out2j[i,j]-out2s[i,j]),2);
end;
aad:=sqrt(aad/((datapoint1-datapoint)*n));
edit20.text:=floattostr(aad);
for  i:=1 to m do
  for j:=1 to r do
  memo13.lines.append(floattostr(w01[i,j])+'  '+inttostr(j)+'  '+inttostr(i));
for k:=1 to n do
  for j:=1 to r do
  memo14.lines.append(floattostr(w12[j,k])+'  '+inttostr(j)+'  '+inttostr(k));
for j:=1 to r do
  memo15.lines.append(floattostr(b1[j])+'  '+inttostr(j));
end;
end.
```