Integrating the Improved CBP Model with Kernel SOM

Qun Dai Songcan Chen*

Department of Computer Science and Engineering, Nanjing University of Aeronautics & Astronautics, Nanjing 210016, China

Abstract. In this paper, we first design a more generalized network model, Improved CBP, based on the same structure as Circular BackPropagation (CBP) proposed by Ridella et al. The novelty of ICBP lies in: 1) it substitutes the original extra added node with the isotropic quadratic form input in CBP with the one with an anisotropic quadratic form input; 2) particularly, the weights between the extra node and all the hidden nodes are endowed fixed values instead of the original changeable values. As a result, ICBP possesses better generalization and adaptability although it has less adjustable weights compared to CBP. Secondly, we propose a new kernel-based SOM algorithm using the kernel method. Our main motives of using the kernel method are 1) to induce a class of robust non-Euclidean distance measures for the original input space and establish a new objective functions for SOM, and thus make the newly-established SOM able to cluster the non-Euclidean structures in data; 2) to enhance robustness of the SOM algorithms to noise and outliers and at the same time still retain computational simplicity. And then, with the combined BP-SOM idea of Weijters, we construct a new integrated network ICBP-KSOM. Our motivation of presenting the integration is to construct a high performance classifier by utilizing both ICBP's good generalization and adaptability and KSOM's higher classification performance and robustness comparing to SOM. Finally, the experimental results on three benchmark data sets show the superiority and effectiveness of our new integration in terms of the t-test.

Keywords: Neural networks; Circular back-propagation neural network (CBP); Improved circular back-propagation neural network (ICBP); Self-organizing feature maps (SOM); Kernel-based self-organizing maps (KSOM); BP-SOM; Classification; t-test.

1 Introduction

EBP (Error Back-propagation) [11] is probably the most popular learning algorithm in the study of artificial neural networks, while multiple-layer perceptron (MLP)[1] has widely received attention in both theory and applications due to its excellent properties like universal approximating ability to arbitrary continuous functions. However, BP has its serious limitations in generalizing knowledge from certain types of learning material [12, 30]. As a case in point, Weijters *et al* recorded that BP

* Corresponding author: Tel: +86-25-84892452; Fax: +86-25-84498069; email: s.chen@nuaa.edu.cn (S.C. Chen); daiqun@nuaa.edu.cn (Q. Dai).

often suffers from overfitting, i.e., it specializes on the input-output mappings of the training instances with a reduced ability to generalize to new instances. The sparseness and high-degree non-linearity of training material are two main causes for overfitting. Although BP networks are basically able to represent any non-linear mapping [1], they are not guaranteed to learn such mappings. Date-calculation task is such an example, i.e., to calculate the day of the week associated with a given date. Norris [12] claims that BP algorithm can only be successful on the date-calculation task when it is decomposed into three subtasks by a human expert. Without human assistance, "the net was able to associate training dates with the days they fell on, but was unable to generalize to new dates"[12]. To overcome some of BP limitations, Weijters *et al* described a new combination of an artificial neural network architecture and a learning algorithm, called BP-SOM [7,8]. "The aim of the BP-SOM learning algorithm is to guide learning in multi-layered feedforward network (MFN) in such a way that the hidden-unit activation patterns produced by the same class become more similar to each other." [7] To achieve this aim, Weijters *et al* combined the BP learning scheme used to train a MFN, with Self-Organizing Maps (SOMs) [4,23]. Each hidden layer of the MFN is associated with one SOM. During training of the MFN, the corresponding SOMs are respectively trained on the hidden-unit activation patterns. In addition to BP, information from these SOMs is also used to update the connection weights of the MFN. Weijters verified through experiments that the BP-SOM network and learning algorithm had a better generalization performance under the same training condition as the BP network. The effect of including SOM information in the error signals is that the hidden-unit activation patterns of instances associated with the same class tend to become increasingly similar to each other [7,8].

Almost in parallel with Weijters BP-SOM, Sandro Ridella *et al* proposed a circular BP neural network (CBP) [15-17] through adding an extra node to the original BP input layer and taking the sum of all squared components of an input vector presented to the network as a incoming signal of the added node. The authors proven that CBP possesses favorable capabilities in generalization and adaptability compared to the MLP model [15-17]. Under the CBP framework, both the vector quantization (VQ) [10,16] and the radial basis function (RBF) networks [17] can respectively be constructed, and hence CBP shows great flexibility. However, there also exist several deficiencies in it: 1) The incoming signal of the extra added node only is an isotropic, i.e., an equally-weighted sum of all squared component values, thus it lacks anisotropy among different components for an input vector; 2) Due to such an isotropy, it cannot simulate the famous Bayesian classifier in a more direct way; 3) It requires probably more hidden nodes to approximate any continuous function to arbitrary precision. As a result, redundant parameters may lead to over-fitting, which will lower the generalization capability [15].

To ameliorate CBP defects, we proposed a general improved network model for CBP, for short, ICBP. Actually, ICBP is similar in structure to CBP, but there are two major changes made: a) the incoming signal of the extra node is not an isotropic sum of all squared input components as in CBP

instead of their anisotropic sum; b) more importantly, the weight values between the extra node and all hidden nodes are set to a special common value (in our case, all 1 or all -1) [14,20] rather than usually adjustable parameters as in CBP so that the total number of the adjustable weights in ICBP is probably reduced. So, the newly-constructed model has following characteristics: Firstly, besides inheriting those CBP characteristic of constructive equivalence to VQ and RBF [10,20], ICBP can also model the famous Bayesian classifier in a direct constructive way [21]. Secondly, although having less adaptable weights than CBP, ICBP has better generalization and adaptation [14,20]. Thirdly, it can still adopt the BP learning algorithm to perform training with the learning complexity equal to that of CBP. Naturally, various existing improved algorithms to BP can also be applied to upgrade performances of CBP and ICBP. In addition, due to assigning special constant values either +1s or –1s to all the weights connecting the added node and all the hidden nodes to respectively form ICBP+1 or ICBP-1 networks, consequently ICBPs have less adjustable weights but better generalization and adaptability than CBP [14,20]. This indeed demonstrates rationality of the famous Occam's razor principle, i.e., network with simple structure but just good training performance is generally better generalization than the one with slightly better training performance but more complex structure [1].

Despite all the meliorations of CBP and ICBP to BP, we find through experiments that neither of them solve highly non-linear classification problem such as Date-Calculation task to a satisfactory degree, although their performances are obviously better than BP. And we also observe by experiments that even BP-SOM still cannot deal with the task satisfactorily, which inspires an idea to further improve BP-SOM architecture and our ICBP network model. As stated above, the effect of combining SOM with BP is that the hidden-unit activation patterns of instances associated with the same class tend to become increasingly similar to each other [7,8]. Therefore, we deduce intuitively that if SOM is combined with ICBP, then ICBP hidden-unit activation output of the same class will tend to become similar. Desirably, ICBP-SOM will also inherit ICBP good performance in generalization and adaptability, which results in a further improvement to BP-SOM classification capability. This is our first alteration to BP-SOM, we substitute the MFN block in BP-SOM with our ICBP.

Next, we introduce kernel method to the integrated architecture. Classical SOMs clustering and classification algorithm developed by Kohonen is performed in the input space based on the Euclidean norm. It fails when input patterns distribution or their inherent structure is highly nonlinear. The kernel methods [3,5,6,9] are one of the most researched subjects within machine-learning community in recent years and have been widely applied to pattern recognition and function approximation. Typical examples are support vector machines [3,5,6,18], kernel Fisher linear discriminant analysis (KFLDA) [26], kernel principal component analysis (KPCA) [9], kernel perceptron algorithm [27], just name a few. The fundamental idea of the kernel method is to transform the original low-dimension inner product input space into a higher-dimension (possibly infinite)

feature space through some nonlinear mapping. In the higher-dimension feature space, complex nonlinear problems in the original low-dimension space can more likely be linearly treated and solved according to the well-known Cover's theorem [28]. However, such a mapping will undoubtedly lead to an exponential increase of computational time, i.e., so-called curse of dimensionality. It is very fortunate that adopting kernel functions to substitute an inner product in the original space is a favorable option, which corresponds to mapping the space into higher-dimension feature space exactly. Therefore, the inner product form leads us to applying the kernel methods to classical SOMs. However, compared to the approaches presented in [29], a major difference of proposed KSOMs in [19, 32] in this paper is: we do not adopt so-called dual representation for each centroid, i.e., a linear combination of all training data. Instead, we directly transform all the centroids in the original space, together with all training data, into high-dimension feature space with an (implicitly) mapping. Such a direct transformation results in two benefits:

1) inducing a class of robust non-Euclidean distance measures for the original input space and establish a new objective function for SOM, and thus make the newly-established SOM able to cluster the non-Euclidean structures in data;
2) enhancing robustness of the SOM algorithms to noise and outliers and at the same time still retain computational simplicity;

Provided with the above two benefits of our KSOM, an idea of combining ICBP with KSOM becomes very clear. The non-Euclidean distance measures of KSOM will be favorable for ICBP-KSOM to deal with the data with non-Euclidean structures, while the high robustness of KSOM will upgrade the whole ICBP-KSOM robustness to outliers and noise. In short, such an integration renders them more useful than either.

In addition, our integration of ICBP with KSOM differs greatly from the general neural network ensemble [33,34], which is an active research area in recent years. Such two types of integrations aim both to improve generalization ability but use different means in achieving this goal. The method of neural network ensemble is training a finite number of neural networks and then combining their results, whereas our manner of integration is completely different with aiming at *just* assisting to train *single* ICBP and BP rather than a set of ICBPs or BPs. Specifically, we integrate ICBP with KSOM through adding the information from KSOMs to ICBP connection weights adjustments, by which the hidden-unit activation patterns of the same class samples in ICBP hidden layers tend to become "increasingly similar" to each other [7,8,23]. After training, the KSOMs associated to each ICBP hidden layer are abandoned and do not participate in the final classification procedure, in other words, a final classification decision is made just by so-trained single ICBP. Thus, ICBP-KSOM belongs to a novel type of integration methodology of BP-SOM [7,8] proposed by Weijters *et al.*

The remaining of this paper is organized as follows: In section 2, we introduce our ICBP network model. KSOM is discussed in section 3. We demonstrate the combination of ICBP and KSOM in section 4. Finally, Section 5 presents our experimental comparisons of BP-SOM and our integrated

network ICBP-KSOM on three benchmark classification tasks, viz. the date-calculation task, the parity-12 task, the task of detecting splices in DNA sequences. In these experiments, our ICBP-KSOM model is shown to greatly improve classification and generalization performance over that of BP-SOM in a sense of the t-test. From these experiments, we draw a conclusion as a whole in section 6 that the new integrated architecture ICBP-KSOM successfully combines advantages of our ICBP and KSOM learning algorithms.

2 ICBP network

Fig. 1 ICBP three-layer network model

ICBP network has entirely the same structure of CBP as shown in Fig.1. It has N_O output nodes, N_h hidden nodes, d input nodes corresponding to d dimensional input pattern. The difference is it has an extra input node: $x_{d+1} = \sum_{i=1}^{d} a_i^2 x_i^2$, while in CBP $x_{d+1} = \sum_{i=1}^{d} x_i^2$ instead. Obviously, if all a_i are assigned equal values, ICBP degrades to CBP. And furthermore, ICBP weights connecting the extra node to all the hidden nodes are taken differently from CBP ones: $w_{j(d+1)}^1 (j = 1, \cdots, N_h)$ for ICBP are assigned a common constant directly while the counterparts are adaptable parameters. Consequently, there exists a discrepancy of $|N_h - d|$ for the number of adaptable parameters of these two models. Generally speaking, the number of hidden nodes is more than input nodes according to the well-known theorem that MFN with sufficient hidden nodes can approximate any continuous function to arbitrary precision [1]. Therefore, the adjustable parameters of ICBP are often less than CBP. However, a valuable character of ICBP is that although it has less adaptable weights, it is better in generalization and adaptability than CBP [14,20].

For convenience, below listed are some notations used in this chapter:

$(x_0, x_1, \cdots, x_d)$: ICBP network input.

x_{d+1}: the extra input as defined above.

η: the ICBP learning rate.

α': the ICBP momentum term.

$V_i^2 = \sum_{j=0}^{N_h} w_{ij}^2 V_j^1, i = 1, \cdots, N_o$: the *i*-th output of ICBP network.

ζ_i^μ: the *i*th element of the desired output vector for pattern μ.

$V_j^1 = g(r_j) = 1 / (1 + e^{-r_j}), j = 1, \cdots, N_h$: the activation output of the *j*th hidden node,

where r_j denotes the following sum of weighted inputs:

$$r_j(\vec{x},\vec{w}^1)=\sum_{i=0}^{d} w_{ji}^1 x_i + w_{j(d+1)}^1 x_{d+1} = -w_{j0}^1 + \sum_{i=1}^{d}(w_{ji}^1 x_i + w_{j(d+1)}^1 a_i^2 x_i^2) \quad (1)$$

From Eq.(1), if all $w_{j(d+1)}^1$ are assigned +1, then ICBP extends the spherical activation fields of CBP hidden neurons to quadratic hyper-ellipsoidal; if $w_{j(d+1)}^1$ are set as +1 or –1 alternatively, then the activation field of ICBP hidden neurons is hyperboloid. Actually different assignments of +1 or –1 to $w_{j(d+1)}^1$ ($j=1...N_h$) are totally 2^{N_h} and thus result in 2^{N_h} different ICBP networks.

The sum-of-squares error function is defined as:

$$E=\frac{1}{2}\sum_{i=1}^{N_o}(\zeta_i^{\mu}-V_i^2)^2 \quad (2)$$

Adopting BP learning algorithm (in fact, any other improved algorithms can be applied), the weight adjustments between output and hidden layer are easily derived as follows:

$$\Delta w_{ij}^2(t+1)=-\eta\frac{\partial E_p}{\partial w_{ij}^2}=\eta(\zeta_i^{\mu}-V_i^2)\cdot w_{ij}^2(t)\cdot V_j^1+\alpha'\Delta w_{ij}^2(t)$$

$$i=1,\cdots,N_o,\ j=0,\cdots,N_h \quad (3)$$

And the adjusting quantities in the weights between the hidden and input layers are:

$$\Delta w_{ij}^1(t+1)=\eta\delta_i^1 V_j^0+\alpha'\Delta w_{ij}^1(t)$$

$$=\eta\delta_i^1 x_j+\alpha'\Delta w_{ij}^1(t) \quad (4)$$

where $\delta_i^1=[\sum_{l=1}^{N_o}(\zeta_l^{\mu}-V_l^2)w_{lj}^2]\cdot V_j^1(1-V_j^1)$, $i=1,\cdots,N_h, j=0,\cdots,d$ (5)

Finally, corresponding formula for $\Delta a_k,(k=1,\cdots,d)$ are:

$$\Delta a_k=2\eta\sum_{j=1}^{N_h}\{\sum_{i=1}^{N_o}[(\zeta_i^{\mu}-V_i^2)w_{ij}^2]\cdot V_j^1(1-V_j^1)\cdot w_{j(d+1)}^1\}\cdot a_k x_k^2,\quad k=1,\cdots,d \quad (6)$$

3 KSOM [19, 32]

The Self-Organizing Map was developed by Kohonen [4] and is based on unsupervised competitive learning. Fig.2 shows the structure of a two-layer SOM. Based on the classical Kohonen SOM formulations [4], we construct the kernel SOM algorithm and modify its objective function with the mapping Φ as follows:

$$J(w_j) = \left\| \Phi(X) - \Phi(w_j) \right\|^2 \tag{7}$$

where $\Phi : x \to \Phi(x) \in F$, $x \in X$, X is patterns set and F is mapped feature space.

Now through the kernel substitution, we have:

$$\begin{aligned} \left\| \Phi(X) - \Phi(w_j) \right\|^2 &= \Phi(X)^T \Phi(X) + \Phi(w_j)^T \Phi(w_j) - 2\Phi(X)^T \Phi(w_j) \\ &= K(X,X) + K(w_j, w_j) - 2K(X, w_j) \end{aligned} \tag{8}$$

in this way we obtain a new class of non-Euclidean distance measures in original data space . To seek the minimum of function $J(w_j)$, we can choose an arbitrary initial point w_0. Start from w_0 and track along the negative gradient direction $s = -\nabla J(w_0)$. For each point w_j, we define an unit vector in the negative gradient direction as:

$$\hat{s}^{(j)} = -\frac{\nabla J(w_j)}{\left\| \nabla J(w_j) \right\|} \tag{9}$$

Then, adjusting formula for w_j based on gradient descent method is:

$$w_j(n+1) = w_j(n) - \rho_j \frac{\nabla J(w_j)}{\left\| \nabla J(w_j) \right\|} \tag{10}$$

Let new learning rate be $\eta'(n) = \eta(n) \dfrac{\left\| \nabla J(w_j) \right\|}{\rho_j}$.

Substitute (8) into (10), we obtain the weight adjustment formula of KSOM:

$$w_j(n+1) = w_j(n) - \eta'(n) \nabla J(w_j) = w_j(n) - \eta'(n) \left(\frac{\partial K(w_j, w_j)}{\partial w_j} - 2 \frac{\partial K(X, w_j)}{\partial w_j} \right) \tag{11}$$

In this way, we derive kernelized SOM algorithm strictly following gradient descent method. Unlike other kernelized algorithms such as SVM and KPCA using the dual representation form for the solution vectors, in our derivation, we conceptually employ some non-linear mapping to transform those patterns, together with those weight vectors associated with the neurons, from low-dimensional space to high-dimensional one. Therefore, the whole learning process of our KSOM is still performed in the original space and thus making the weight vectors be explicitly represented in the original space. Moreover, based on the flexibility of kernel mappings, different kernel functions lead to different distance measures, depending on those concrete problems to be solved. In this paper, we adopt the following three classical kernel functions which satisfying Mercer condition:

(1) RBF kernel $K(x,y) = e^{-\frac{\|x-y\|^2}{2\sigma^2}}$ (12)

(2) polynomial kernel (POLY) $K(x,y) = (x^T \cdot y)^d, d \geq 2$ (13)

(3) logarithm kernel (LOG) $K(x,y) = \log(1 + \left\|x-y\right\|^2 / \sigma^2)$ (14)

Respectively, substitute (12) into (11),

$$w_j(n+1) = w_j(n) - \eta'(n) \cdot 2/\sigma^2 \cdot e^{-\left\|X-w_j\right\|^2/\sigma^2} \cdot (X - w_j) \quad (15)$$

Substitute (13) into (11),

$$w_j(n+1) = w_j(n) - \eta'(n)(2d(w_j^T)^{d-1} w_j - (X^T w_j)^{d-1} X) \quad (16)$$

Substitute (14) into (11),

$$w_j(n+1) = w_j(n) + \eta'(n) \cdot \frac{1}{\sigma^2 \cdot (1 + \left\|X - w_j\right\|^2 / \sigma^2)} \cdot (X - w_j) \quad (17)$$

The above equations (15), (16), (17) are weight adjustment formulae of the KSOMs based on the three kernel functions. When $\sigma \to \infty$, KSOM is degenerated to SOM, i.e., SOM can be taken as a special case of KSOM.

It can be seen that in KSOM algorithm, although we replace the inner product in pattern space with kernel function, in essence, clustering still proceeds in the original pattern space. The difference is: Euclidean distance are no longer the measure between patterns and weight vectors, instead kernel functions are employed. Herein, we generalize SOM classifier base on Euclidean distance to several kinds of new classifiers based on different kernel-induced distance measures in the same space.

4 ICBP-KSOM Neural Network Model

ICBP-KSOM is a hybrid neural network architecture combining ICBP model and KSOM, simulating Weijters developing BP-SOM [7,8], as shown in Fig.3. The ICBP-KSOM architecture adds a kernel Self-Organizing Map to each hidden layer of the ICBP. During training the weights of ICBP, the corresponding KSOM is trained on the hidden-unit activation patterns. In addition to ICBP error, the information from the KSOMs is taken into account when updating the ICBP connection weights. Through introducing KSOM information into the ICBP error signals, the hidden-unit activation patterns of the same class samples tend to become “increasingly similar” to each other [7,8,23]. The extending from BP-SOM to ICBP-KSOM is driven by the thought of utilizing both ICBP’s good generalization and adaptability and KSOM’s good classification performance and robustness [19, 32] to create a high quality classifier.

Fig. 3 The BP-SOM Architecture

The purpose of using KSOM is to help to train the ICBP by influencing the $\delta' s$ of the hidden layers. Each KSOM is trained with the kernel SOM algorithm using the activation values V_i^m of its corresponding hidden layer. After several training epochs the KSOM achieves some degree of self-organization. This self-organization information is used to compute a *ksom_error* for each KSOM,

which is used to influence the ICBP $\delta' s$, before they are used to update the weights. Class label and reliability properties [as defined in (27)] are extended for each KSOM to help determining whether the *ksom_error* should be considered. As a whole, the ICBP-KSOM algorithm consists of two parts [7,8,23]:

1. The learning algorithm of the combined network based on ICBP model.
2. The KSOM updating algorithm to determine the class labels and reliabilities.

The following step-by-step procedure shows how the algorithm works. In this section, we will use the notations listed below:

ξ_k^μ : The *k*-th input of pattern μ .

V_i^m : The *i*th unit output of ICBP layer *m*.

w_{ij}^m : The connecting weight from the *j*-th neuron of layer *m-1* to *i*-th neuron of layer *m*.

δ_i^m : The error signal of unit *i* in ICBP layer m.

M: The output layer has $m = M$.

h_i^m : The input for *i*th unit of ICBP layer *m*, $h_i^m = \sum_j w_{ij}^m V_j^{m-1}$, where *m*>0.

α : The influence of *ksom_error* vector on the error signal of ICBP hidden layer nodes.

r: The reliability of KSOM winning unit *i**.

t: The threshold parameter of KSOM winning unit *i**, typically setting at 0.95 to prevent unreliable KSOM elements to influence the error signals of ICBP.

4.1 The Learning Algorithm of ICBP-KSOM

1. Initialize the ICBP weights to small random values.
2. Choose a pattern ξ^μ and apply it to ICBP input layer so that for all *k*

$$x_k = \xi_k^\mu \text{, and } x_{d+1} = \sum_{k=1}^{d} a_k^2 x_k^2 \tag{18}$$

3. Propagate the signal forward through the network using

$$V_i^m = g(h_i^m) = g(\sum_j w_{ij}^m V_j^{m-1}) \tag{19}$$

for each *i* and *m* until the final outputs V_i^M have all been calculated.

And ICBP weights connecting the extra node to all the hidden nodes: $w_{j(d+1)}^1,\ j = 1, \cdots, N_h$ are assigned a common constant. And in this work, they are all assigned as –1's.

4. After propagating an input sample through the ICBP, the activation values of the hidden layers

are used as training inputs for the KSOM's, which are trained following the kernel SOM algorithm discussed in Section 3. Next, a winner i^* is selected from each KSOM, which has the same class label as the input pattern. Then an error vector v^m is computed using

$$v_j^m = w_{i^* j}^m - V_j^m \tag{20}$$

for each node *j* in *m*-th hidden layer.

5. Compute the deltas for ICBP output layer:

$$\delta_i^M = g'(h_i^M)[\zeta_i^\mu - V_i^M] \tag{21}$$

6. Compute the deltas for the ICBP preceding layers by propagating the errors backwards:

$$\delta_i^{m-1} = g'(h_i^{m-1}) \sum_j w_{ij}^m \delta_j^m \tag{22}$$

7. Apply the error vector computed from KSOM to update the deltas:

$$\delta_i^m = \begin{cases} (1-\alpha)\delta_i^m + r\alpha v_i^{m-1} & if\ r > t \\ \delta_i^m & otherwise \end{cases} \tag{23}$$

8. Update all connection weights according to $w_{ij}^m(t+1) = w_{ij}^m(t) + \Delta w_{ij}(t)$, where

$$\Delta w_{ij}^m(t+1) = \eta \delta_i^m V_j^{m-1} + \alpha' \Delta w_{ij}^m(t) \tag{24}$$

As for three-layer ICBP network model applied in this work, we use Eq.(3), (4) and (6) to update the weight adjustments Δw_{ij}^2 between output and hidden layer, Δw_{ij}^1 between the hidden and input layers, and $\Delta a_k, (k = 1, \cdots, d)$, respectively. Whereas δ_i^1, δ_i^2 can be derived according to Eq.(21), (22) and (23), with the results shown in Eq.(3) and (5).

9. Turn to step 2 and repeat for the next pattern until all patterns have been processed.

4.2 The KSOM updating algorithm

1. Initialize all the label counters to zero.
2. Choose a pattern ξ^μ and apply it to ICBP input layer so that for all *k*

$$x_k = \xi_k^\mu, \text{ and } x_{d+1} = \sum_{k=1}^{d} a_k^2 x_k^2 \tag{25}$$

3. Propagate the signal forward through the network using

$$V_i^m = g(h_i^m) = g(\sum_j w_{ij}^m V_j^{m-1}) \tag{26}$$

for each *i* and *m* until the final outputs V_i^M have all been calculated.

4. Apply the activation values of the hidden layers to the accompanying KSOM. Increase the label counter of KSOM element winner by one for the label of the current pattern.
5. Turn to step2 and repeat for the next pattern until all patterns have been presented.
6. After all patterns have been propagated through ICBP-KSOM, choose the final class label and reliability as defined in (27) for each KSOM element according to:
   - The label with the biggest count is selected as the class label for the KSOM element.
   - Compute the corresponding reliability as follows:

$$reliability = \frac{counting\ times\ of\ the\ label}{total\ class\ labels\ counting\ times} \qquad (27)$$

5 Experiment Results

In order to compare our ICBP-KSOM with BP-SOM, we present experimental results on three benchmark classification tasks (viz. the data-calculation task, the parity-12 task and the task of detecting splices in DNA sequences). In these experiments, three classical kernel functions were applied to BP-KSOM and ICBP-KSOM, i.e., RBF kernel, polynomial kernel and logarithm kernel. The BP or ICBP learning rate was set to 0.25 and the momentum to 0.4. α was set to 0.25, and the reliability threshold t to 0.95 . Class labeling was performed at each 5th cycle (Weijters, 1995; Weijters *et al*, 1997).

5.1 Date Calculation Task [7,8,23]

The date calculation task is the problem to classify dates (e.g., April 6, 1997) to the day of the week on which they fall. It is an example of task which easily leads to overfitting in MFNs trained by BP. Norris described the problem in his paper and concluded that BP was not able to learn this task, unless it was decomposed into three easier subtasks [7,8,12]. The dates used to train and test the two models were chosen from July 1, 1970 to April 1, 2004. The training set consists of 4110 instances, and the test set consists of 1000 new instances. The input for the networks consists of 85 inputs representing the year (35 units), the month (12 units) and the day of the month (31 units). The developers of the BP-SOM algorithm tested BP-SOM by comparing its performance with that of normal BP. In order to compare our three new models discussed earlier with BP-SOM, we applied them to the date calculation task respectively. We tested the two models using a hidden layer with 15, 20 and 40 units and a SOM or a kernel-based SOM with 10x10 2D grid SOM output layer. We used 20 different random initializations and training was stopped at 2000 epochs. The average classification errors and error variances are given in Table 1 and Table 4, respectively. Mostly, the average error rates and error variances of our ICBP-KSOMs are obviously less than those of CBP, ICBP and BP-SOM, except that the error variance of logarithm kernel (LOG) ICBP-KSOM is bigger than that of ICBP with 15 hidden nodes. However, the average error rates and error variances of LOG ICBP-KSOM with 20 or 40 hidden nodes all show significantly better generalization performances over CBP, ICBP and BP-SOM.

Therefore, the network structure can influence ICBP-KSOM performance heavily, while for this simulation task, ICBP-KSOM with 40 hidden nodes exhibits the best generalization capability.

Insert Table 1 here

5.2 12-Parity Task [7,8,23]

The 12-Parity task is the problem to determine whether a pattern of 12 0's and 1's contains an even number of 1's. The training set contained 1,000 different instances selected at random out of the 4096 (2^{12}) possible patterns of length 12. The test set contained 100 new instances. There were four different networks tested using hidden layers of 15, 20, 40 units, respectively. The SOM or KSOM used for the four integrated networks has a size of 10X10. Each network configuration performed 20 testing runs using 20 different random initializations. The average classification errors and error variances after 1000 epochs are shown in Table 2 and 4. The results again indicate better classification and generalization performances of our algorithm.

Insert Table 2 here

5.3 Gene Detection [7,8,24]

A third comparative experiment was performed using the gene benchmark data sets extracted from the Proben1 benchmark collection [24]. The data set detects intron/exon boundaries (splice junctions) in nucleotide sequences. From a window of 60 DNA sequence elements (nucleotides), we can decide whether the middle is either an intron/exon boundary (a donor), or an exon/intron boundary (an acceptor), or none of these. The data set features 3000 training instances and 190 test instances. The MFN or ICBP used in the experiments contained 120 input units, 10 hidden units and 3 output units (representing 'intron-exon boundary', 'exon-intron boundary', or 'neither'). Here, the size of the corresponding SOM or KSOM was set to 3×3. Table 3 and 4 displays the classification results of the four models, again indicating advantages of our ICBP-KSOM network over CBP, ICBP, and BP-SOM.

Insert Table 3 here

5.4 T-Test [31]

To find out whether ICBP-KSOM is better than the other three models, we carry out t-test on the classification results of the 20 runs for the above three simulation tasks for statistically significant differences between ICBP-KSOM (RBF, POLY, LOG) and CBP, ICBP, BP-SOM, respectively. The null hypothesis H_0 demonstrates that there is no significant difference between the mean number of patterns misclassified by ICBP-KSOM and CBP, ICBP or BP-SOM. The t-values are listed in Table 5, from which we can see that in most cases the hypothesis H_0 are rejected at the 5% significance level (t-value >= 1.686), which means our ICBP-KSOMs possess significantly superior classification

capabilities to the other models. The case of acceptance to the hypothesis H_0 appears mostly in the comparison to ICBP in the Parity-12 task. From Table 2, 4 and 5, we find that it is relatively easy to classify the Parity-12 task and all the four types of models perform closely to each other in this simulation. For this particular application, ICBP model represents a similar classification performance, which can be viewed as a special case.

A remark: We notice that in both this and the first experiments, the numbers of ICBP hidden nodes are both less than those of input nodes, which contradicts our claims in Section 2. In fact, the number of hidden nodes is, in theory, greater than those of the input nodes due to the proven result that a forward multi-layer networks with sufficiently large hidden node number can approximate any continuous function to arbitrary precision. However, in practice, such a conclusion does not necessarily always hold because of only the finite training samples available.

6 Conclusions

Retaining the original structure of BP-SOM, we obtain a new network ICBP-KSOM by introducing our ICBP network model and KSOM to BP-SOM architecture and using the similar way of combination. On the one hand, our ICBP network has an interesting property that although it may have less adaptable weights, it is better in generalization and adaptability than CBP. On the other hand, our KSOM possesses high reliability and robustness for linear inseparable classification problems. Therefore, our thought of combining them together following the same way as BP-SOM is natural and rational. The results of benchmark classification experiments and t-test prove the feasibility and effectiveness of our new integrated algorithm. Such an integration renders them more useful than either. The ICBP-KSOM network architecture showed better classification and generalization performances in various experiments, viz. date-calculation, parity-12, and gene boundary detection. In general, we conclude that the hybrid ICBP-KSOM architecture and learning algorithm successfully combines advantages of our ICBP and KSOM models.

Acknowledgements

We thank Jiangsu Natural Science Key project (BK2004001), Jiangsu “QingLan” Project Foundation and the Returnee’s Foundation of China Scholarship Council for partial supports respectively.

References

[1] H. Simon, Neural Networks, A Comprehensive Foundation, Published by Prentice-Hall, Inc., (1999).

[2] J.A. Hertz, A.S. Krogh & R.G. Palmer, Introduction to the theory of neural computation, Addison-Wesley, (1991).

[3] N. Cristianini and J.S. Taylor, An introduction to SVMs and other kernel-based learning methods, Cambridge Univ. Press, (2000).

[4] T. Kohonen, Self-organisation and Associative Memory, Berlin, Springer Verlag, (1989).

[5] V. N. Vapnik, The nature of statistical learning theory, Springer Verlag, (1995).

[6] V. N. Vapnik, Statistical learning theory, Wiley, (1998).

[7] A. Weijters, The BP-SOM architecture and learning rule, Neural processing letters, 2(6) (1995) 13-16.

[8] A. Weijters, V. D. Bosch, H. J. Herik,. Intelligible neural networks with BP-SOM, Marcke and Daelemans, (1997) 27-36.

[9] B. Scholkopf, A.J. Smola & K.R. Muller, Nonlinear component analysis as a kernel eigenvalue problem, Neural Computation, 10(5) (1998) 1299-1319.

[10] B.Z. Zhang and S.C. Chen, Equivalence between vector quantization and ICBP networks. Journal of Data Acquisition and Processing (in Chinese) 16(3) (2001) 291-294.

[11] D.E. Rumelhart, G. E. Hinton & R. J. Williams, Learning internal representations by error propagation, Parrallel Distributed Processing: Explorations in the Microstructure of Cognition 1: Foundations, (1986) 318-362.

[12] D. Norris, How to build a connectionist idiot (savant). Cognition, 35 (1989) 277-291.

[13] L. Zhang, W. D. Zhou, & L.C. Jiao, Kernel clustering algorithm, Chinese Journal of Computers, 25(6) (2002) 587-590.

[14] Q. Dai, S. C. Chen, & B. Z. Zhang, Improved CBP neural network model with applications in time series prediction, Neural Processing Letters, 18 (2003) 197-211.

[15] S. Ridella, S. Rovetta & R. Zunino, Circular back-propagation networks for classification, IEEE Transactions and Neural Networks , 8(1) (1997) 84-97.

[16] S. Ridella, S. Rovetta & R. Zunino, Circular back-propagation networks embed vector quantization, IEEE Transactions and Neural Networks 10(4) (1999) 972-975.

[17] S. Ridella, S. Rovetta & R. Zunino, CBP networks as a generalized neural model, Int. Conf. Neural Networks (1997)

[18] V. Vapnik and C. Cortes, Support vector networks, Machine Learning, 20 (1995) 273-297.

[19] Z. S. Pan, S. C. Chen & D. Q. Zhang, A kernel-based SOM classification in input space, Acta Electronica Sinica, 32(2) (2004) 227-231.

[20] B. Z. Zhang, The research on the performance and applications of improved BP neural networks, Thesis of Master degree, Computer Science Department of Nanjing University of Aeronautics and Astronautics, Feb. 2001.

[21] B. Z. Zhang, S. C. Chen, The equivalence between ICBP and the Bayesian classifier, Tech. Report No. 021, Dept. of Computer Science & Engineering, Nanjing University of Aeronautics and Astronautics, 2001.

[22] D. MacDonald, C. Fyfe: The kernel self-organizing map, http:// cis.paisley.ac.uk/ fyfe-ci0/kernel/kmeans.ps

[23] J. Eggermont, Rule-extraction and learning in the BP-SOM architecture, Thesis of Master degree,

Computer Science Department of Leiden University, Aug. 1998.

[24] L. Prechelt, Proben1: A set of neural network benchmark problems and benchmarking rules. Technical Report 24/94, Fakultat fur informatic, University Karlsruhe, Germany, 1994.

[25] S. C. Chen, D. Q. Zhang, Robust Image Segmentation Using FCM Kernel-Induced Distance Measure, IEEE Transactions on Systems, Man, and Cybernetics—part B: Cybernetics, 34(4) (2004) 1907-1916

[26] V. Roth, V. Steinhage, Nonlinear discriminant analysis using kernel functions, Neural Information Processing Systems A Solla, T. K. Leen, and K.-R. Muller, Eds. Cambridge, MA: MIT Press 12, (2000) 568–574.

[27] J. H. Chen, C. S. Chen, Fuzzy kernel perceptron, IEEE Trans. Neural Networks, 13 (2002) 1364–1373.

[28] T. M. Cover, Geomeasureal and statistical properties of systems of linear inequalities in pattern recognition, Electron. Comput., 14 (1965) 326–334.

[29] M. Girolami, Mercer kernel-based clustering in feature space, IEEE Trans. Neural Networks, vol. 13 (2002) 780–784.

[30] S. Weiss, C. Kulikowski, Computer Systems that Learn, San Mateo, CA: Morgan Kaufmann. (1991)

[31] R. Dutta, E. L. Hines, J. W. Gardner & P. Boilot, Bacteria classification using cyranose 320 electronic nose, Biomedical Engineering Online, 1(4) (2002) 1-7.

[32] P. Andras, Kernel-Kohonen networks, International Journal of Neural Systems, 12(2) (2002) 117-135.

[33] Z. H. Zhou and S. F. Chen, Neural network ensemble, Chinese Journal of Computers, 25(1) (2002) 1-8.

[34] Z. H. Zhou, et al., View-invariant face recognition based on neural network ensemble, Chinese Journal of Computer Research & Development, 38(10) (2001) 1204-1210.

Table 1. The average classification error on the test set using 20 different random initializations. The network consisting of 15, 20, 40 hidden units was trained for 2000 epochs on the Date Calculation task.

Hidden nodes	CBP	ICBP	BP-SOM	ICBP-KSOM		
				RBF	POLY	LOG
15	23.28	11.25	11.57	8.98	9.84	7.48
20	17.26	9.64	8.96	4.09	4.45	3.84
40	9.71	5.90	6.43	3.84	2.02	2.12

Table 2. The average classification error on the test set using 20 different random initializations. The network consisting of 15, 20, 40 hidden units was trained for 1000 epochs on the 12-Parity task.

Hidden nodes	CBP	ICBP	BP-SOM	ICBP-KSOM		
				RBF	POLY	LOG
15	6.72	3.29	4.67	2.93	3.37	2.77
20	5.67	2.72	3.35	1.98	2.27	1.87
40	4.57	1.84	2.71	1.30	1.59	1.06

Table 3. The average classification error on the test set using 20 different random initializations. The network consisting of 10 hidden units was trained for 200 epochs on the gene detection task.

Hidden nodes	CBP	ICBP	BP-SOM	ICBP-KSOM		
				RBF	POLY	LOG
10	37.51	10.45	10.67	7.90	8.46	9.46
20	26.26	9.30	10.43	7.43	7.29	6.87
40	19.03	8.90	9.88	5.65	4.39	6.87

Table 4. The error variance after 20 tests running on the three simulation tasks, where Hn represents the number hidden nodes utilized in each model.

Simulations	Hn.	CBP	ICBP	BP-SOM	ICBP-KSOM		
					RBF	POLY	LOG
Date Calculation	15	56.55	5.07	11.32	4.62	4.57	11.15
	20	36.56	12.95	5.57	5.0	0.88	1.33
	40	29.25	5.60	9.53	2.36	1.05	0.54
12-Parity Task	15	4.56	7.14	8.38	2.35	3.68	4.42
	20	7.33	4.35	4.40	0.70	2.05	0.68
	40	6.26	1.12	3.92	0.82	2.36	0.61
Gene Detection Task	10	125.70	8.32	8.98	2.07	7.08	8.42
	20	123.05	5.44	9.35	3.55	4.19	4.43
	40	41.41	7.05	6.51	4.39	1.13	9.14

Table 5. Comparison using t-test of CBP, ICBP, BP-SOM and ICBP-KSOM with RBF, POLY and LOG kernels, respectively.

Simulations	Hn	CBP			ICBP			BP-SOM		
		RBF	POLY	LOG	RBF	POLY	LOG	RBF	POLY	LOG
Date Calculation	15	7.96	7.49	8.37	3.18	1.99	4.07	2.82	1.90	3.76
	20	8.90	9.12	9.50	5.70	6.08	6.69	6.52	7.73	8.49
	40	4.56	6.09	6.06	3.18	6.55	6.64	3.28	5.90	5.91
12-Parity Task	15	6.28	5.09	5.75	0.83*	0.39*	0.94*	2.31	1.63*	2.31
	20	5.67	4.84	5.85	1.43*	0.77*	1.65*	2.63	1.85	2.85
	40	5.35	4.43	5.83	1.67*	0.58*	2.56	2.50	1.95	3.37
Gene Detection Task	10	11.41	10.99	10.56	3.44	2.21	1.05*	3.62	2.40	1.26*
	20	7.29	7.33	7.49	2.73	2.82	3.38	3.64	3.71	4.18
	40	8.62	9.78	7.45	4.19	6.88	2.20	5.58	8.65	3.31

The asterisk (*) indicates the difference between the two corresponding models is not significant at 5% significance level, i.e. t-value < 1.686.

Captions of figures:

Fig. 1 ICBP three-layer network model

Fig. 2 SOM network structure

Fig. 3 The BP-SOM Architecture

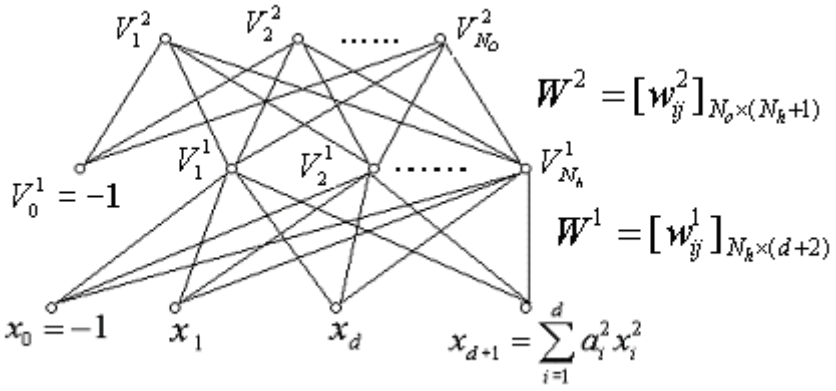

Fig. 1 ICBP three-layer network model

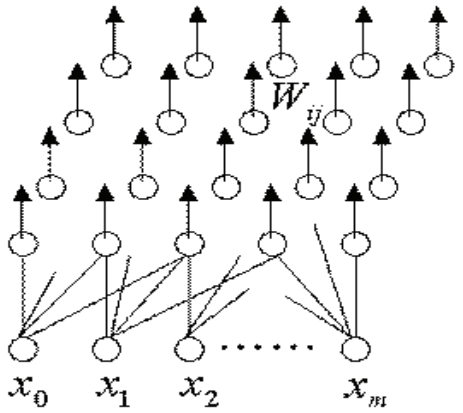

Fig. 2 SOM network structure

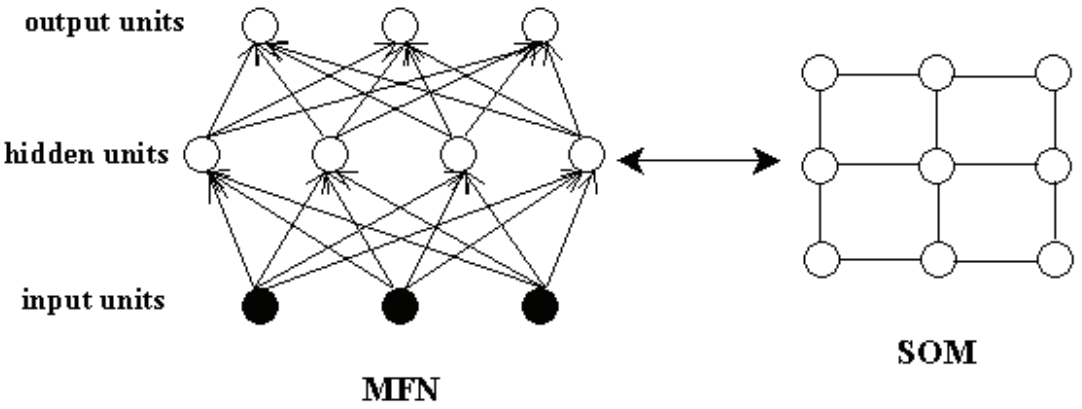

Fig. 3 The BP-SOM Architecture