Classification Using Multi-valued Pulse Coupled Neural Network

Xiaodong Gu

Department of Electronic Engineering, Fudan University, Shanghai 200433, China
guxiaodong@263.net, xdgu@fudan.edu.cn

Abstract. This paper introduces how to use multi-valued PCNN (Pulse Coupled Neural Network) proposed in this paper to do classification. 2-dimensional data can be projected onto two-dimensional PCNN locally laterally linked. Different pulse waves generated by training data label different regions corresponding to different classes. The same pulse wave labels the region corresponding to the same class. Meeting of different pulse waves obtains the separatrixes of different classes. In order to differentiate different pulse waves, outputs of neurons in PCNN should be multi-valued. We call networks composed of these neurons multi-valued PCNNs. The number of classes determines the number of output value of each neuron. *N*-valued PCNN can be used to classify *N-1* different classes. Experimental results of the 2-dimensional salmon-weever classification show that the correct recognition rate of test set is 98.11% (3477/3544) when training samples are only 10% of all samples.

Keywords: PCNN (Pulse Coupled Neural Network); Multi-valued PCNN; Pulse waves; Classification.

1 Introduction

Assigning labels to data based on their features is the function of the classifier. In this paper, we use pulse-spreading of multi-valued PCNN (Pulse Coupled Neural Network) that we proposed based on conventional binary PCNN to assign labels. In 1990, Eckhorn introduced the linking network [1] exhibiting synchronous pulse bursts in the cat or the monkey visual cortex [2]–[4]. It is a model pulse-emitting and spatio-temporal coding. Introducing the linking strength to the linking model obtains binary PCNN [5], which retains the main characteristics of the linking network.

Binary PCNN has been applied in many fields, such as image processing, object detection, optimization [6]-[18]. Almost all existing applications of binary PCNN are based on its pulse-spreading characteristic. Using this pulse-spreading characteristic can extend binary PCNN's applications conveniently. The output of each neuron in binary PCNN is binary. In this paper multi-valued PCNN, consisting of neurons with multi-valued outputs, is introduced, and pulse-spreading of multi-valued PCNN is used in classification. Pulses generated by conventional binary PCNN have the same amplitude value so that using them cannot label different classes. So pulses generated by conventional binary PCNN cannot be used to do classification. If amplitudes of pulses are multi-valued, multi-valued amplitudes can be used to label different

M. Ishikawa et al. (Eds.): ICONIP 2007, Part II, LNCS 4985, pp. 549 – 558, 2008.

classes. Therefore, in this paper we introduce multi-valued PCNN to do classification. In classification based on multi-valued PCNN, different values of pulse amplitudes are used to differentiate different pulse waves that label different classes. In classification, firstly 2-dimensional training data are projected onto 2-dimensional multi-valued PCNN locally laterally linked. Then we make neurons that correspond to training data of different classes emit different-valued pulses. These pulses spread over the network in parallel like the flood, and the same-valued pulses generate the same-valued pulse waves. The training data in the same class produce same-valued pulses so as to generate the same-valued pulse waves. Same-valued pulse waves label their spreading-over regions as the same class. In this paper, in order to classify N different classes, the output of each neuron should be $(N+1)$–valued. In training of classification, when different pulse waves corresponding to different classes meet, the separatrixes of different classes are obtained. In test, test data are projected onto the network and they are assigned labels of the regions that training data have already labeled. In this paper, we address this approach in 2-dimensional space.

In Section 2, multi-valued PCNN for classification is described. In Section 3, classification based on multi-valued PCNN is introduced. In Section 4, computer simulations of 2-class and 2-dimensional classification based on 3-valued PCNN, and experimental results of the practical application of 3-valued PCNN to salmon-weever classification are shown, followed by computer simulations of 3-class and 2- dimensional classification based on 4-valued PCNN. Conclusions are obtained in Session 5.

2 Multi-valued PCNN

Fig.1 illustrates a simplified neuron j in multi-valued PCNN in classification in this paper. It consists of three parts: the receptive field, the modulation field, and the multi-valued pulse generator. Its output is multi-valued, not binary, which is different from that in binary PCNN.

Eq. (1) to Eq.(5) describe this neuron. Neuron j has two channels (F channel and L channel). F channel is feeding input ($F_j(t)$); and L channel is linking input ($L_j(t)$).In Fig.1, I_j is an input signal from the external source, and only inputs to F channel of j (see (1)). In classification, if neuron j is projected onto a training sample, I_j equals to

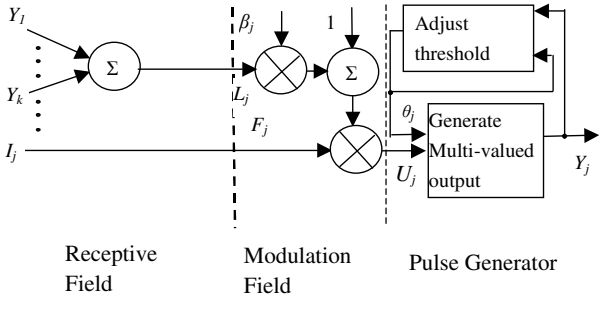


Fig. 1. A simplified neuron j in multi-valued PCNN in this paper

the value corresponding to this training sample's class label. If neuron j is not projected onto a training sample, I_j equals to a smaller value. $Y_1(t),\ldots,Y_k(t)\ldots$, multi-valued output signals of neurons connected with j , are also input signals of neuron j, and only input to L channel of j. $Y_j(t)$ is the output signal of neuron j. In (2), $N(j)$ is the neighbor field of neuron j and j may belong or not belong to $N(j)$. In this paper, j does not belong to $N(j)$. $L_j(t)$is added a constant positive bias firstly. Then it is multiplied by $F_j(t)$ and the bias is taken to be unity (see Eq. (3)). In Eq. (3), β_j is the linking strength. The total internal activity $U_j(t)$is the result of modulation and it inputs to the multi-valued pulse generator. If $U_j(t)$ is greater than the threshold $\theta_j(t)$, the neuron output $Y_j(t)$ equals to a value determined by $(U_j(t)-\theta_j(t))$. Then $Y_j(t)$ feedbacks to make $\theta_j(t)$ rise over $U_j(t)$immediately so that $Y_j(t)$ turns into 0. Therefore, when $U_j(t)$is greater than $\theta_j(t)$, neuron j outputs a multi-valued pulse.

Eq.(4) shows the 3-valued output case. In Eq.(4), C_1, C_2 and b_1, are constants, and according to the algorithm in this paper, we can assign them suitable values easily. 3-valued PCNN composed of 3-valued neurons can be used to classify 2 different classes. N-valued PCNN composed of N-valued neurons can be used to classify N-1 different classes. In conventional binary PCNN neuron, if $U_j(t)$ is greater $\theta_j(t)$, $Y_j(t)$ equals to 1; else equals to 0. It is the main difference between multi-valued PCNN in this paper and conventional binary PCNN that whether the output of each neuron is multi-valued. If neuron j outputs a multi-valued pulse (namely Y_j is more than 0), we call neuron j firing. In Eq.(5), V_j^T and α_j^T are the amplitude gain and the time constant of the threshold adjuster. Solving Eq.(5), the lower limit of integration is just before the last firing. When neuron j fires, its threshold increases to constant V_j^T fast, and the threshold value is independent of the threshold value at the time just before firing. Then $\theta_j(t)$ descends with time increasing.

$$F_j(t) = I_j \tag{1}$$

$$L_j(t) = \sum_{k \in N(j)} Y_k(t) \tag{2}$$

$$U_j(t) = F_j(t)[1 + \beta_j L_j(t)] \tag{3}$$

$$Y_j(t) = \begin{cases} C_2 \quad , \quad b_1 \le U_j(t) - \theta_j(t) \quad , \textit{firing} \\ C_1 \quad , 0 < U_j(t) - \theta_j(t) < b_1 \quad , \textit{firing} \\ 0 \quad , \quad U_j(t) - \theta_j(t) \le 0 \quad , \textit{non} - \textit{firing} \end{cases} \tag{4}$$

$$\frac{d\theta_j(t)}{dt} = -\alpha_j^T + V_j^T Y_j(t) \tag{5}$$

In 2-dimensional classification, multi-valued PCNN is a single layer 2-dimensional array of laterally linked neurons and one of those neurons is illustrated in Fig.2. All neurons are identical. Each neuron is connected with neurons by its L channel in its 4-neighbor field (See Fig.2). In 2-dimensional classification each datum corresponds

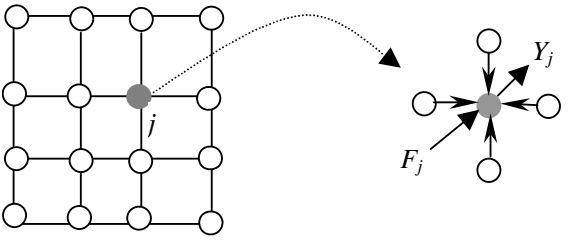


Fig. 2. The connection mode of each neuron in multi-valued PCNN for 2-dimensional classification

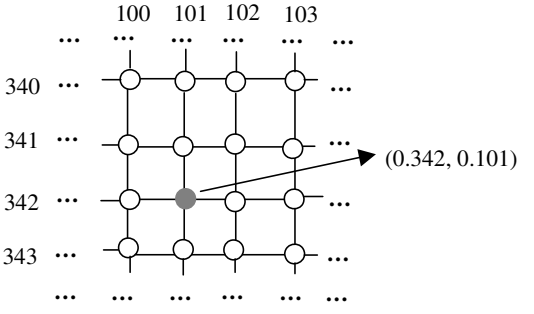


Fig. 3. The neuron in the 342nd row and the 101st column corresponds to the 2-dimensional data sample (0.342, 0.101)

to a neuron in 2-dimensional PCNN. A 2-dimensional PCNN likes a coordinate plane and neurons in different locations can denote different data. In this paper, 2-dimensional data are normalized in [0.001,1.000], and the effective value is millesimal (0.001), and the size of PCNN is 1000*1000 (the number of neurons is million). For example, if a data sample is (0.342, 0.101), it is projected onto neuron (342, 101) in multi-valued PCNN (see Fig.3). Namely the neuron in the 342nd row and the 101st column corresponds to the datum (0.342, 0.101).

In training, if a neuron is projected onto a training sample, the *F* channel of this neuron receives the value '*l*' corresponding to this sample's class label, namely $F_j(t) = I_j = l$; if a neuron is not projected onto a training sample, the feeding input ($F_j(t)$) equals to an other value smaller than those corresponding to class labels. These values can not equals to 0, or pulses cannot spread over the network. For 2-class classification, $F_j(t) = I_j, I_j \in \{l_1, l_2, l_3\}$, l_1, l_2, l_3 are constants corresponding to non-training-sample, the training sample of class I, and the training sample of class II respectively. We can assign l_1, l_2, l_3 suitable values easily based on the algorithm in this paper. In Section 4, the parameter values of multi-valued PCNN in our experiments are shown. After training, each neuron is assigned a label. In test, test samples are projected onto multi-valued PCNN labeled, and labels of test samples equal to those of corresponding neurons.

3 Classification Using Multi-valued PCNN

Pulse-spreading of binary PCNN has been efficiently used in image processing and optimization. For example, we have used binary PCNN in image thinning [10]. In thinning, skeletons are obtained when pulse waves meet. In training in classification in this paper, first data are projected onto multi-valued PCNN. More corresponding details have been shown in Section 2. Next, multi-valued pulse-waves generated by training samples label all regions, and separatrixes are obtained when different pulse waves meet. In thinning based on binary PCNN, pulse waves are identical. In classification based on multi-valued PCNN, pulse waves are not identical, and multi-valued PCNN is used to generate different pulse waves to label different classes. *N*-valued PCNN can be used to differentiate *N-1* different classes. For example, *4*-valued PCNN can be used to differentiate *3*different classes.

In training in classification, neurons corresponding to training data samples fire to produce pulses to excite neurons not projected onto training data samples to generate multi-valued pulse-waves to spread over the network in parallel. Setting enough large threshold amplitude gain (V_j^T make each neuron fire only once. Output values of firing neurons captured by different pulse-waves are different. The output value of the firing neuron labels this neuron. When all neurons have fired, the training finishes. After training, each neuron is assigned a class label. In test, test data samples are projected onto labeled PCNN, and their labels are those of corresponding neurons. Now, we describe the multi-valued PCNN classification algorithm. Firstly we introduce the symbols in the following algorithm. **F** is a feeding input matrix. In training, if a neuron is projected onto a training sample, the corresponding element in **F** equals to the value corresponding to the class label of this training sample. **L** is a linking input matrix, where $\mathbf{L}= \mathbf{Y} \otimes \mathbf{K}$, where **Y** is a multi-valued output matrix, and '$\otimes$' indicates two-dimensional convolution, and **K** is a 3*3 kernel,

$$\mathbf{K}= \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

This **K** corresponds to the connection mode of 4-neighbor field. Because in this paper each neuron's 4-neighbor field does not include itself, the center element of **K** equals to 0. **U** is an internal activity matrix and **Θ** is a threshold matrix. **Lab**, a label matrix, saves the training result, namely the segmentation result of the classification space. **F**, **L, U, Y**, **Θ,** and **Lab** have the same dimensions (1000*1000). β is the linking strength and each neuron has the same β. *Height* (1000) and *Width*(1000) are the height and the width of 2-dimensional multi-valued PCNN.

The training process of multi-valued PCNN classification algorithm for 2-dimensional data is described below.

1) 2-dimensional training samples are projected onto 2-dimensional multi-valued PCNN.
2) **L = 0.U = 0. Y = 0.** β=1. Initialize **F, Θ** and other parameters.
3) Calculate **L ,U ,Y** in order**.**

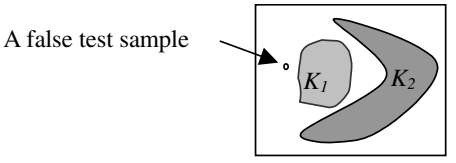


Fig. 4. In this 2-class (K_1 andK_2) classification space, all data distribute in shadows. A false test sample out of shadows is not rejected and still assigned the label corresponding to class K_1, by using multi-valued PCNN classification algorithm.

4) If **Y**(i,j) >0, record the corresponding class label to **Lab**(i,j) and increase **Θ**(i,j) to a enough large value so that the corresponding neuron will not fire again. (i=1,…,*Height*; j=1,…*Width*)
5) If all neurons have fired**,** end; else go back to 3)

In test, projecting test samples onto **Lab** obtains test results quickly. Pulse-spreading is only used in training and not be used in test. Using **Lab** can be easy to obtain separatrixes too. In 2-dimensional classification, separatrixes are 2-dimensional curves. In this algorithm, any test sample is assigned a class label and a false test sample not belonging to any existing class is also assigned a label. For example (see Fig.4), in a 2-class (K_1 andK_2) classification space, all data (the training and the test data) distribute in part regions of the whole classification space and these distribution regions (namely data sets) are described by shadows in Fig.4. In Fig.4, a false test sample out of shadows appears, and it is incorrectly assigned the class label corresponding to class K_1 although it neither belongs to class K_1 nor belongs to class K_2 . In order to reject false test samples, in training we can finish the training before all neurons fire by controlling the spreading distances of pulse waves. However, in this situation, the gaps might appear in shadows in Fig.4. If correct test samples falls into these gaps, they will not be assigned class labels and rejected incorrectly. In this paper, training does not finish until all neurons have fired. Therefore, at last any test sample is assigned a label. The parameter values of multi-valued PCNN in our experiments are shown in Section 4.

4 Experimental Results and Discussions

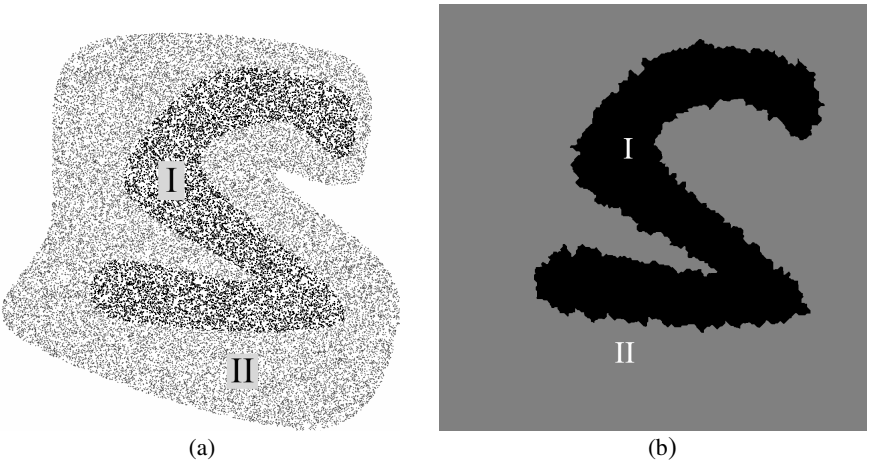
4.1 Computer Simulations of 2-class and 2-dimensional Classification Based on 3-valued PCNN and the Practical Salmon-weever Classification

In computer simulations of 2-class and 2-dimensional classification based on 3-valued PCNN, we used 50 data sets, one of which is shown in Fig.5 (a). For each data set, data are normalized in [0.001,1.000] and the effective value is millesimal (0.001), and the size of PCNN is 1000*1000 (the number of neurons is 1 million). For each data set, we have randomized 10% of all samples as training ones, and used residuary samples as test ones. The mean correct recognition rate of test set is 97.77% and that of training set is 100%.

In initialization of the feeding input matrix **F** for 2-class and 2-dimensional classification, if a element corresponds a training sample, its value is 2 for class I and 10 for

class II; if a element does not correspond to a training sample, its value is 1. Namely, $F_j(t) = I_j, I_j \in \{1,2,10\}$. At the beginning, each element of threshold matrix Θ equals to 1.5. When a neuron fires, increase its threshold to an enough large value (1000) so that it will not fire again. Each neuron's linking strength (β) is 1. Eq.(6) is the output function of each neuron. According to the algorithm, it is easy to choosethe parameters of the neuron.

$$Y_j(t) = \begin{cases} 10 & , \quad 8.5 \le U_j(t) - \theta_j(t) \quad , firing \\ 2 & , 0 < U_j(t) - \theta_j(t) < 8.5 \quad , firing \\ 0 & , \quad U_j(t) - \theta_j(t) \le 0 \quad , non - firing \end{cases} \tag{6}$$



(a) (b)

Fig. 5. (a) A data set among 50 data sets in computer simulations of 2-class and 2-dimensional classification based on 3-valued PCNN. There are 36587 samples in this set. (b) The training results based on 10% of all samples in (a). The training samples are randomizedïïIn (b), the black region corresponds to class I and the grey region corresponds to class II. In test, each test sample is assigned the class label according to which region it is projected onto.

In the practical application, we used 3-valued PCNN to classify 2 different kinds of fish (salmons and weevers). In general, weever is lighter than salmon and its width is also some different from salmon's width, so the mean lightness and the width of the fish were used as 2 features in classification. The range of the width (12cm-25cm) is normalized in [0.001,1.000]. The mean lightness (0-255) is also normalized in [0.001,1.000]. After normalization, a sample can be projected onto a neuron in 3-valued PCNN (1000*1000). In this salmon-weever classification, there are 2135 salmons and 1703 weevers (see Fig.6(a)). We randomized 394 ones, 10% of all samples, as training ones, and used residuary 3544 samples as test ones. In the test set, 3477 test samples are assigned correct labels and 67 test samples are labeled wrong labels. The correct recognition rate of test set is 98.11% (3477/3544).

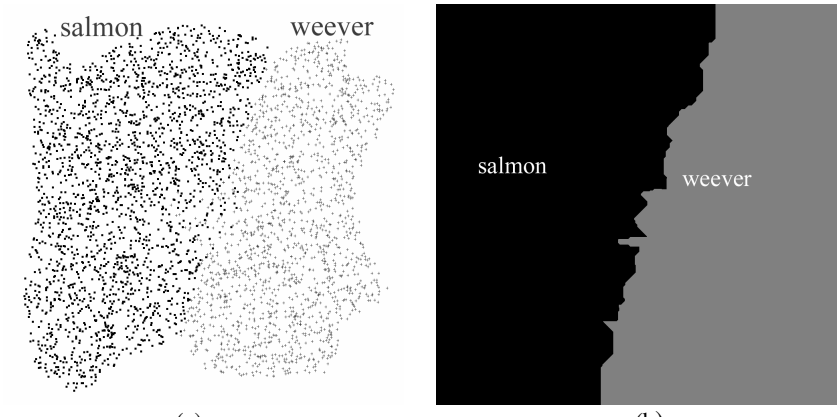


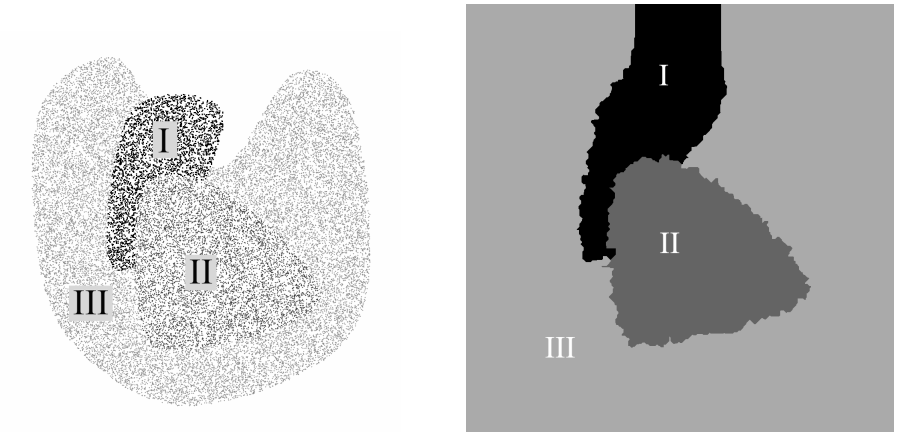
Fig. 6. A practical 2-class and 2-dimentional salmon-weever classification based on 3-valued PCNN. The 2-dimensional feature vector is (*width, lightness*). The mean lightness increases from the left to the right horizontally, and the width increases from the top down vertically in (a) or (b). (a) All 3938 samples (2135 salmons and 1703 weevers). (b) The training results based on 394 samples, 10% of all samples in (a) and the training samples are randomizedïïIn (b), the black region corresponds to class salmon and the grey region corresponds to class weever. In test, each test sample is assigned the class label according to which region it is projected onto, and the correct recognition rate of residuary test samples is 98.11%.

4.2 An Example of 3-class and 2-dimensional Classification Based on 4-valued PCNN

Fig. 7 illustrates an example of 3-class and 2-dimensional classification using 4-valued PCNN. In this example data are also normalized in [0.001,1.000], and the effective value is also millesimal (0.001), and the size of PCNN is also1000*1000. In initialization of the feeding input matrix **F**, if a element corresponds a training sample, its value is 2 for class I, 10 for class II, and 50 for class III; if a element does not correspond to a training sample, its value is 1. $F_j(t) = I_j, I_j \in \{1,2,10,50\}$. At the beginning, each element of threshold matrix $\boldsymbol{\Theta}$ equals to 1.5. When a neuron fires, increase its threshold to an enough large value (20000) so that it will not fire again. Each neuron's linking strength (β) is 1. Eq. (7) is the output function of each neuron for this 3-class and 2-dimensional classification. This output function is 4-valued.

$$Y_j(t) = \begin{cases} 50, & 48.5 \le U_j(t) - \theta_j(t), & ,firing \\ 10, & 8.5 \le U_j(t) - \theta_j(t) < 48.5 & ,firing \\ 2, & 0 < U_j(t) - \theta_j(t) < 8.5 & ,firing \\ 0, & U_j(t) - \theta_j(t) \le 0 & ,non-firing \end{cases} \tag{7}$$

This method also can be extended to 3-dimensional classification. In 3-dimensional classification, separatrixes are 3-dimensional hood faces. If we want to extend this



(a) (b)

Fig. 7. An example of 3-class and 2-dimensional classification. (a) All 28684 samples in the data set (class I, class II, and class III). (b) The training result based on random 10% of all samples in (a). In (b) the black region corresponds to class I, and the dark grey region corresponds to class II, and the light grey region corresponds to class III. In test, each test sample is assigned the label according to which region it is projected onto and the correct recognition rate of residuary test samples is 99.23%.

method to *k*-dimensional classification (k>3, $k \in Z$), in simulations we may connect neurons based on indexes of multi-arrays where they are stored. However, it is very complex to do so, and pulse-spreading phenomena are not so obvious as in 2 or 3-dimensional classification.

5 Conclusions

Pulse-spreading of binary PCNN can be extended to multi-valued PCNN, and multi-valued pulse waves of multi-valued PCNN can be applied in classification. In multi-valued PCNN classification, data samples are projected onto multi-valued PCNN, and different multi-valued pulse waves label different classes in parallel. Meeting of different multi-valued pulse waves obtains the separatrixes of different classes. *N*-valued output can be used to classify *N-1* different classes. In this paper, we address this method in 2-dimensional classification. Experimental results of 2-dimensional classification show that multi-valued PCNN in this paper can efficiently do classification even the training samples are only 10% of all samples. In the salmon-weever classification, the correct recognition rate of test set is 98.11% (3477/3544) when training samples are only 10% of all samples. Next we will use more practical data to test this method further and try to extend this method to higher dimensional space.

Acknowledgments. This work was supported by National Natural Science Foundation of China (No. 60671062 and 60571052) and National Basic Research Program of China (2005CB724303).

References

1. Eckhorn, R., Reitboeck, H.J., Arndt, M., et al.: Feature Linking via Synchronization among Distributed Assemblies: Simulation of Results from Cat Cortex. Neural Computation 2, 293–307 (1990)
2. Eckhorn, R., Bauer, R., Jordan, W., et al.: Coherent oscillations: a mechanism of feature linking in the visual cortex? Multiple electrode and correlation analyses in the cat. Biological Cybernetics 60, 121–130 (1988)
3. Gray, C.M., Konig, P., Engel, A.K., Singer, W.: Oscillatory responses in cat visual cortex exhibitioner-columnar synchronization which reflects global stimulus properties. Nature 338, 334–337 (1989)
4. Eckhorn, R., Frien, A., Bauer, R., et al.: High frequency oscillations in primary visual cortex of awake monkey. NeuroRep. 4, 243–246 (1993)
5. Johnson, J.L., Ritter, D.: Observation of Periodic Waves in a Pulse-coupled Neural Network. Opt. Lett. 18, 1253–1255 (1993)
6. Johnson, J.L., Padgett, M.L.: PCNN Models and Applications. IEEE Trans. on Neural Networks 10, 480–498 (1999)
7. Kuntimad, G., Ranganath, H.S.: Perfect Image Segmentation Using Pulse Coupled Neural Networks. IEEE Trans. on Neural Networks 10, 591–598 (1999)
8. Broussard, R.P., Rogers, S.K., Oxley, M.E., Tarr, G.L.: Physiologically Motivated Image Fusion for Object Detection using a Pulse Coupled Neural Network. IEEE Trans. on Neural Networks 10, 554–563 (1999)
9. Kinser, J.M.: Foveation by a Pulse-Coupled Neural Network. IEEE Trans. on Neural Networks 10, 621–625 (1999)
10. Gu, X.D., Yu, D.H., Zhang, L.M.: Image Thinning Using Pulse Coupled Neural Network. Pattern Recognition Letters 25, 1075–1084 (2004)
11. Gu, X.D., Yu, D.H., Zhang, L.M.: Image Shadow Removal Using Pulse Coupled Neural Network. IEEE Trans. on Neural Networks 16, 692–698 (2005)
12. Caulfield, H.J., Kinser, J.M.: Finding Shortest Path in the Shortest Time Using PCNN's. IEEE Trans. on Neural Networks 10, 604–606 (1999)
13. Gu, X.D., Guo, S.D., Yu, D.H.: A New Approach for Automated Image Segmentation Based on Unit-linking PCNN. In: Proceedings in IEEE International Conference on Machine learning and Cybernetics, Beijing, China, pp. 175–178 (2002)
14. Gu, X.D.: A New Approach to Image Authentication Using Local Image Icon of Unit-linking PCNN. In: Proceedings in Int. Joint Conf. Neural Networks, Vancouver, Canada, pp. 2015–2020 (2006)
15. Gu, X.D., Zhang, L.M., Yu, D.H.: General Design Approach to Unit-linking PCNN for Image Processing. In: Proceedings in Int. Joint Conf. Neural Networks, Montreal, Canada, pp. 1836–1841 (2005)
16. Gu, X.D., Zhang, L.M.: Global Icons and Local Icons of Images based Unit-linking PCNN and their Application to Robot Navigation. In: Wang, J., Liao, X.-F., Yi, Z. (eds.) ISNN 2005. LNCS, vol. 3497, pp. 836–841. Springer, Heidelberg (2005)
17. Gu, X.D.: Research on Pulse Coupled Neural Network and its Applications. Ph.D. dissertation, Peking University (2003)
18. Gu, X.D.: Research on several Theoretical and Applied Aspects on Unit-linking Pulse Coupled Neural Network. Post-doctoral research report, Fudan University (2005)