A Sub-dimension Based Probabilistic Neural Network for Occlusion Classification

Yi Wang, Benson S. Y. Lam, and Hong Yan

***Abstract*—The development of DNA microarray technology has made it possible for scientists to study the expression of a large number of genes at the same time under various conditions. One problem in microarray data analysis is to deal with occlusion. In real world datasets, some features of gene expression may be partly missing or corrupted easily, causing a wrong determination using existing classification methods. In this paper, we adopt a sub-dimension based probabilistic neural network to solve this problem. Compared with original probabilistic neural networks, experiment results show that the proposed method achieves a better performance.**

***Index Terms*—Microarray data clustering, probabilistic neural network, sub-dimension method, occlusion problem.**

I. Introduction

The development of the microarray technology has made it possible to monitor gene expressions for tens of thousands of genes in parallel and enhance our understanding of functional genomics. An important task in DNA microarray data analysis is to identify genes which have similar expression patterns in order to understand their biological functions and cellular processes. This process could be done manually, in which case the amount of effort would be tremendous and intensive. Thus, it is important to develop computerized data analysis techniques, such as classification algorithms, which is needed in many applications. However, microarray data can become partly corrupted easily because of the imperfect mechanics of conducting physical tests, such as dye inconsistencies and slight mechanical differences in measurement equipment [1]. These corruptions can cause significant noise in the measured data and lead to incorrect classifications in data analysis. Because of the randomness of the noise, some measured data values deviate from their true values significantly and become unusable. However, we do not necessarily know which of the measured data values are corrupted. We call this the data occlusion problem. In this paper, we propose a sub-dimension based probabilistic neural network to solve this problem. Our experiment results show that this method is effective.

Manuscript received December 9, 2007. This work was partly supported by the Hong Kong Research Grant Council (Projects CityU 122506).

Y. Wang is with the School of Electrical Information Engineering, University of Sydney, Sydney, NSW 2006 Australia (phone: +61-2-93517070; fax: +61-2-93513847; e-mail: yi@ee.usyd.edu.au).

B. S. Y. Lam is with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong, China (e-mail: benson.lam@student.cityu.edu.hk).

H. Yan is with the Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong, China and the School of Electrical Information Engineering, University of Sydney, Sydney, NSW 2006 Australia, (e-mail: h.yan@cityu.edu.hk).

Probabilistic neural network (PNN) was first developed by D. Specht [2], [3]. It provides a general solution to pattern classification problems by using the Bayes strategy for probability density functions. It is frequently employed in pattern classification and microarray data clustering due to its prominent time efficiency. It provides a considerable improvement in training speed compared to the conventional back-propagation network (BPN). Furthermore, as discussed in [4], PNN could attain the same accuracy as BPN.

We assume that a $n \times d$ matrix X contains d features and one or two features partly missing or corrupted. The corruption in the datasets may cause significant error in conventional clustering methods since all features are considered in every training sample. In the proposed method, we adopt the sub-dimension method combined with the PNN. The sub-dimension method requires the partition of the entire dataset into several small parts called sub-dimensions, which may or may not be disjoint [5]. The proposed method clusters the datasets into their sub-dimensions. We assign x_1 and x_2 to the same group if more than half of the sub-dimensions of x_{1j} and x_{2j} belong to the same group. The experiment results show a better performance with the proposed method that conventional ones.

In this paper, we first briefly review the structure of the PNN and discuss the sub-dimension formulation in Section II. Then we describe the proposed method and present experimental results from five datasets in Sections III and IV. Finally, discussions and conclusions are given in Section V.

II. Theory

A. Probabilistic Neural Network

Instead of using the conventional back-propagation neural network, we adopt the PNN in the proposed method, considering its primary advantages of convenient binary outputs and fast training speed.

PNN is based on the Bayes strategies, which are implemented by a method which minimizes the "expected risk" of misclassification [2], [3]. Considering a two-category situation for instance, the problem is to decide to which classes pattern X would belong, θ_A or θ_B. In this case, the Bayes decision rule is written as follows:

$$\begin{aligned} d(X) &= \theta_A \quad \text{if } h_A l_A f_A(X) > h_B l_B f_B(X) \\ d(X) &= \theta_B \quad \text{if } h_A l_A f_A(X) < h_B l_B f_B(X) \end{aligned} \tag{1}$$

where $f_A(X)$ and $f_B(X)$ are the probability density

functions for categories θ_A and θ_B, respectively; l_A is the loss function associated with the decision $d(X)=\theta_B$ when the truth is θ_A, l_B is the loss function associated with the decision $d(X)=\theta_A$ when the truth is θ_B; h_A and h_B are the a priori probabilities of occurrence of patterns from categories θ_A and θ_B, respectively.

The main task of implementing Equation (1) is to estimate the probability density function for each class according to a set of known training patterns. As in papers [2] and [3], it is shown that a particular estimation of a probability density function of category θ_A is

$$f_A(X)=\frac{1}{(2\pi)^{p/2}\sigma^p}\frac{1}{m}\cdot\sum_{i=1}^{m}\exp\left[-(X-X_{Ai})^t(X-X_{Ai})/(2\sigma^2)\right] \quad (2)$$

where i is the current pattern number; m is the total number of training patterns; X_{Ai} is the $i-th$ training pattern form category θ_A; σ is the smoothing parameter. $f_A(X)$ is the sum of multivariate Gaussian distributions centered at each training sample. It could be any smooth density function, not limited on the Gaussian.

The PNN network consists of input units, two hidden layers, and output units. Fig. 1 shows the PNN structure for a two group classification.

The input units in the PNN correspond to input features. The first hidden layer is called pattern units. In each unit, input pattern X is performed a dot product with a weight vector W_i, $Z_i = X \cdot W_i$, and then a nonlinear operation is implemented. Unlike back-propagation, the sigmoid activation function is replaced by an exponential function which could be represented as follows:

$$\exp\left[(Z_i-1)/\delta^2\right] \quad (3)$$

If both X and W_i are normalized to unit length, Equation (3) becomes:

$$\exp\left[-(W_i-X)^t(W_i-X)/(2\delta^2)\right]. \quad (4)$$

Summation units which are the second hidden layer simply sum the input from the corresponding pattern units according to the training process. The connection between two hidden layers is made in that way that each pattern unit in the first layer matches only one appropriate node in the second layer.

The output units, or decision units, simply produce a binary output, as indicated in Fig. 1.

PNN employs the training patterns to estimate the probability distribution of each class during the training routine, and classifies the input according to the weighted average of the closest training examples in the testing process. In this paradigm, learning for small and moderate sized databases is faster since the iteration process is avoided. However, the entire training datasets need to be stored and large networks require large databases. These are the disadvantage of PNN [6].

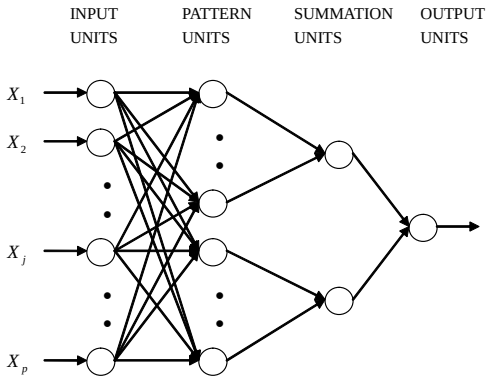

Figure 1. PNN network.

Figure 1. The PNN structure for a two group classification.

B. Sub-dimension

The sub-dimension method [5] can be implemented by dividing the data bases into smaller parts and applying a classification procedure to each part. Let x_{ij} be a matrix with i objects (rows) and j features (columns), and

$$X = \lfloor A_1 \quad A_2 \cdots A_j \cdots A_d \rfloor$$

where $1 \le j \le d$, A_j represents the jth feature of all objects. We redefine

$$X = \lfloor B_1 \quad B_2 \cdots B_p \rfloor$$
$$B_j = \lfloor A_{j1} \quad A_{j2} \cdots A_{js} \rfloor$$

where $p \le d$, s represents the number of features in each sub-dimension and $s \le d$. Now X is expressed by a set of overlapping sub-dimensions B_j.

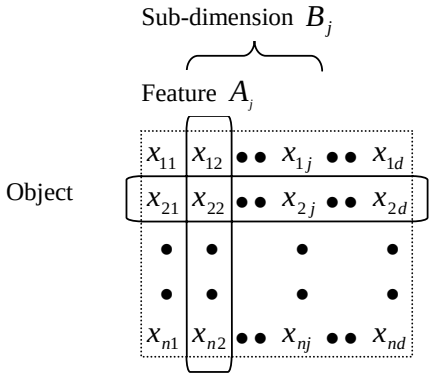

Figure 2. The sub-dimensions of dataset X.

Instead of considering all features as evidence for classification, the sub-dimension based algorithm respects the sub-dimension B_j as input pattern and implements a classification method to each sub-dimension respectively. The observable benefit of this approach is that results of each sub-dimension are not affected by features in other sub-dimensions.

We conclude that object X_1 is closer to X_2 than X_3 when more than half of the sub-dimensions $X_{1(Bj)}$ are closer to $X_{2(Bj)}$ than $X_{3(Bj)}$. This can be formulated by:

$$Card\left(\left\{j:\left\|X_{1(B_j)}-X_{2(B_j)}\right\|<\left\|X_{1(B_j)}-X_{3(B_j)}\right\|\right\}\right) > Card\left(\left\{j:\left\|X_{1(B_j)}-X_{2(B_j)}\right\|\geq\left\|X_{1(B_j)}-X_{3(B_j)}\right\|\right\}\right) \quad (5)$$

where $Card(S)$ refers to the cardinality (or the number of elements) of the set S [5].

In this paper, we simply employ the majority decision as the class label determination. We assign object x_i to a group, if a majority of sub-dimensions x_{ij} are classified to that group.

III. SIMULATION METHOD

We assume that the original $n \times d$ dataset

$$X = \begin{bmatrix} x_1 & x_2 \cdots x_j \cdots x_d \end{bmatrix}$$

where each dimension x_j (where $1 \leq j \leq d$) has n objects. Since the purpose of the proposed method is to solve the occlusion problem, we add white Gaussian noise (wgn) into one of the dimensions of datasets as corruption. In this paper, we add noise in different dimensions for different datasets.

$$X = \begin{bmatrix} x_1 & x_2 \cdots (x_j + wgn) \cdots x_d \end{bmatrix}$$

The learning session is important for PNN which affects the response of the network directly. We partition the objects (rows) into two parts randomly: the training set X_{ta} for training the network and the testing set X_{te} for testing the proposed method.

Then, both X_{ta} and X_{te} were divided into p sub-dimensions (columns). We denoted them as Y_p.

$$X = \begin{bmatrix} Y_1 & Y_2 \cdots Y_p \end{bmatrix}$$

where $p \leq d$ and each sub-dimension has s features.

We adopt the sub-dimension Y_1 in the training set X_{ta} as the input pattern for the PNN training and obtain a classifier. Then we use the classifier to determine the class label of the testing set X_{te} . We reiterate the process for all sub-dimensions by using the same random order training set X_{ta} and testing set X_{te} . Thus, after all sub-dimension network clustering, we obtain a set of class labels in which every testing object has p class labels from p classifiers.

To combine the p class labels into one, we vote for each testing object and choose the majority as the class label. In the last step, we compare the results of proposed method with the known results to calculate the accuracy.

IV. RESULTS

Experiments based on the proposed method are performed on three real world datasets: iris data, wine data, Wisconsin diagnostic breast cancer (wdbc) data, and two microarray datasets: yeast cell cycle data [7], and sporulation data [8]. For each dataset, we run the steps in section III 30 times and employ their average and standard derivation (STD) to show the performance. The normal PNN is adopted for comparison.

A. Real world data

The iris dataset includes 150 objects in three groups with four features. We add the noise in the first dimension as corruption and 50 objects are selected as training samples and the remaining 100 objects are used for testing. Table 1 shows that the proposed method achieves an improvement of 3% in recognition accuracy that the conventional PNN.

The wine dataset contains 178 objects in three groups and 13 features. In our experiment, we adopt 78 objects as training samples and the remaining 100 objects for testing. The noise is added in the seventh dimension which is the middle of all the features. As shown in Table 2, the proposed method obtains 93 correct out of 100, compared with 72 correct out of 100 in normal PNN. We can see that the proposed method provides a significant improvement in accuracy for this classification problem.

The wdbc dataset has 576 objects in two classes and 30 features in which 276 training samples and 300 testing samples are used to test the recognition results. The last dimension becomes corrupted by Gaussian noise. As in the case for the iris and wine data, the proposed method shows better results in the wdbc dataset, 281 correct classifications compared with 265 by the conventional PNN.

For all the three datasets, the STD is lower compared to the normal PNN method. This shows that the consistency of our method is comparatively better than the normal PNN method.

B. Microarray data

Two microarray datasets are used in our experiments: the yeast dataset and the sporulation dataset. For both of these the proposed method shows a better performance.

The yeast cell cycle dataset consisting of 6220 genes is published by Cho et al. [7]. In the study of the sub-dimension method [5], we adopt 384 genes and normalized each gene expression profile so that it has zero mean and unit variance. The dataset has five cycle phases which are the G1 phase, late G1 phase, S phase, S2 phase and M phase, and 17 time points, in which the first dimension is corrupted. The results are given in Table 4. The proposed method correctly classifies 150 out of 200 testing samples with the standard derivation 5.0 and the conventional PNN correctly classifies 142 with a standard derivation of 5.6.

The sporulation dataset contains 6118 genes with seven features. In [5], after pre-processing, we only adopt 1136 genes of which the value of the root mean square of the log2 transformed the data greater than 1.13. The dataset has seven phases: metabolic, early I, early II early middle, middle, mid-late, and late. We corrupt the last dimension and adopt 736 genes as training and the remaining 400 genes for testing. As shown in Table 5, the proposed method works well with

an accuracy rate 49.5% (198 out of 400) compared with 44% for the conventional PNN. The standard derivation of the result from the proposed method is larger than for the conventional PNN, but this is the only case in all the datasets that has a larger STD.

As shown in the tables, the proposed method performs better than the conventional PNN in all datasets.

V. DISCUSSION AND CONCLUSION

Instead of considering all features of a dataset at the same time, the proposed method partitions the dataset into small parts and implements the PNN for each part of the dimensions at a time. Thus, the corrupted dimension which may cause a wrong classification in one sub-dimension will hardly affect the accuracy of others. Since we choose the majority class that all sub-dimensions represent, the effect of the corrupted dimension can be minimized, and thus the final result of the proposed method is more accurate.

The running time of the proposed method is more than the conventional network because of the repeated network operations for each sub-dimension. However, the fast training speed of the PNN may compensate for this disadvantage.

Microarray data has a strong correlation among the adjoining features. The experiment results show that a large *s* can produce a better accuracy in microarray data analysis. However, the determination of the *s* value still depends on the nature of different experiments. This is an issue that needs to be studied further.

APPENDIX

Table 1. Classification results for the iris data.

Group	IRIS	
	PNN	Proposed method
1	34	35
2	28	30
3	29	28
Total	91	93
%	91%	93%
STD	2.5	1.9

Table 2. Classification results for the wine data.

Grou p	WINE	
	PNN	Proposed method
1	29	33
2	29	34
3	14	26
Total	72	93
%	72%	93%
STD	4.0	2.2

Table 3. Classification results for the wdbc data.

Grou p	WDBC	
	PNN	Proposed method
1	177	181
2	88	100
Total	265	281
%	88.3%	93.7%
STD	4.1	3.2

Table 4. Classification results for the yeast cell cycle.

Phases	YEAST	
	PNN	Proposed method
Early G1	25	26
Late G1	61	64
S phase	15	18
G2	16	18
M phase	25	24
Total	142	150
%	71%	75%
STD	5.6	5.0

Table 5. Classification results for the sporulation data.

Phases	SPORULATION	
	PNN	Proposed method
Metabolic	28	33
Early I	46	55
Early II	37	43
Early middle	24	24
Middle	20	20
Mid-late	2	2
Late	19	21
Total	176	198
%	44%	49.5%
STD	8.3	9.0

ACKNOWLEDGMENT

This work is partially supported by a grant from the Hong Kong Research Grant Council (Project CityU 122506).

REFERENCES

[1] B. Comes and A. Kelemen, "Probabilistic neural network classification for microarraydata," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 3, July 2003, pp. 1714-1717.

[2] D. F. Specht, "Probabilistic neural networks for classification, mapping or associative memory," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 1, San Diego, CA, July 1988, pp. 525-532.

[3] D. F. Specht, "Probabilistic neural networks and the polynomial adaline as complementary techniques for classification," in *IEEE Trans. Neural Networks*, Jan. 1990, pp. 111-121.

[4] D. F. Specht and P. D. Shapiro, "Generalization accuracy of probabilistic neural networks compared with back-propagation networks," in *IJCNN-91-Seattle Int. Joint Conf. Neural Networks*, vol. 1, July 1991, pp. 887-892.

[5] B. S. Y. Lam and H. Yan, " A sub-dimension based similarity measure for DNA microarray data clustering," in *Physical Review E*, Vol. 74, 2006, pp. 041906.

[6] S. V. Kartalopoulos, *Understanding Neural Networks and Fuzzy Logic: Basic Concepts and Applications*, USA: IEEE Neural Networks Council, 1996, pp. 104-105.

[7] R. Cho, M. Campbell, E. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. Wolfsberg, A. Gabrielian, D. Landsman, D. Lockhart, and R. Davis, Molecular Cell 2, 1998, pp. 65-78.

[8] S. Chu, J. DeRisi, M. Eisen, J. Mulholland, D. Botstein, P. Brown, and I. Herskowitz, Science 282, 1998, pp. 699-705.