Bioinformation by Biomedical Informatics Publishing Group *open access*

www.bioinformation.net **Prediction Model**

Entropy based sub-dimensional evaluation and selection method for DNA microarray data classification

Yi Wang[1,]* and Hong Yan[1, 2]

[1]School of Electrical and Information Engineering, University of Sydney, Sydney, NSW 2006 Australia; [2]Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong; Yi Wang* - Email: kingoneonewy@hotmail.com; * Corresponding author

received July 21, 2008; accepted September 13, 2008; published November 03, 2008

Abstract:
DNA microarray allows the measurement of expression levels of tens of thousands of genes simultaneously and has many applications in biology and medicine. Microarray data are very noisy and this makes it difficult for data analysis and classification. Sub-dimension based methods can overcome the noise problem by partitioning the conditions into sub-groups, performing classification with each group and integrating the results. However, there can be many sub-dimensional groups, which lead to a high computational complexity. In this paper, we propose an entropy-based method to evaluate and select important sub-dimensions and eliminate unimportant ones. This improves the computational efficiency considerably. We have tested our method on four microarray datasets and two other real-world datasets and the experiment results prove the effectiveness of our method.

Keyword: DNA microarray; datasets; entropy; sub-dimension; probabilistic neural network

Background:

The development of microarray technology has made it possible to measure the expression levels of 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 can 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 are needed in many applications. In our previous study, we proposed a sub-dimension based probabilistic neural network to solve this problem **[1]**. 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 back-propagation neural network (BPN).

We assume that the input data consist of an n by d matrix X, where n is the number of genes (objects) and d the number of conditions (features). The sub-dimension based method partitions the dataset into several smaller parts called sub-dimensions, which may or may not be disjoint **[5]**. It clusters the datasets based on their sub-dimensions. In our previous study, a voting system was used to combine all sub-dimension class results. We assigned two objects x_1 and x_2 to the same group if more than half of the sub-dimensions x_{1j} and x_{2j} belong to the same group. Experiment results show that the method is effective **[1]**. However, the enormous number of features in the real world microarray datasets makes it difficult to select the optimal sub-dimensions. One method is to reduce the dimensionality. In the classification, the contribution of each sub-dimension is not equal. Some may be corrupted or less relative to others, which can be discarded without degrading the performance of the system. In this paper, we employ the feature evaluation and selection technique to determine the sub-dimensions that are not as important as others in order to reduce the number of sub-dimensions without affecting the classification accuracy.

The aim of feature selection is to discriminate features which contain the most or the least effective information from an original candidate set. Feature selection algorithms have been well researched in this area. In our study, we apply the entropy based measure combined with the sub-dimension method. Entropy based methods have been used in many areas, such as mathematics, communication theory, and economics. In 1948, Shannon **[6]** first introduced the basic entropy and the information gain concept to the information domain. "Entropy is a measure of the amount of uncertainty in the outcome of a random experiment, or equivalently, a measure of the information obtained when the outcome is observed." **[7]** In our study, the entropy can be said to be the measure of contribution that a single sub-dimension makes to the general classification. Aiming to show the convincing performance of the proposed method, normal PNN and sub-dimension combined PNN are used in experimental comparison. In this paper, we first briefly review the structure of the PNN, discuss the sub-dimension formulation, and introduce the entropy concept. Then, we describe the proposed method and present experiment results from six datasets.

Methodology:
Please see supplementary material.

Discussion:
Experiments based on the proposed method are performed on four microarray datasets including yeast cell cycle data, sporulation data, rodrigues data, and annot data **[11]**-**[14]**. To verify the proposed method, we also present the experiment results on other datasets, including wine data, Wisconsin diagnostic breast cancer (wdbc) data. For each dataset, we run the steps in section II 30 times and compute their average to evaluate the performance.

Real world data
In order to evaluate the performance of the proposed method for noisy data, we added white Gaussian noise (wgn) randomly into the features of entire datasets as a form of corruption. 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. As shown in Table 1 (supplementary material), the sub-dimension based PNN obtains 90 correct out of 100, compared with 71 correct out of 100 in normal PNN. However, with 89% accuracy, we can see that the proposed method provides a comparable performance with the sub-dimension based PNN.

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. As in the case for the wine data, the proposed method shows close results in the wdbc dataset, 279 correct classifications compared with 280 by the sub-dimension based PNN, and is superior to the normal PNN.

Microarray data
The yeast cell cycle dataset consisting of 6220 genes is published by Cho and colleagues **[11]**. 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. The results are given in Table 3 under supplementary material. The proposed method correctly classifies 149 out of 200 testing samples and the sub-dimension based PNN correctly classifies 150. The error is only 0.5%.

The sporulation dataset contains 6118 genes with seven features. In **[5]**, after pre-processing, we use only 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 use 736 genes for training and the remaining 400 genes for testing. As shown in Table 4 (supplementary material), the proposed method works well with an accuracy rate of 48.5% (194 out of 400) compared with 49.5% for the sub-dimension based PNN.

Rodriguez dataset is available elsewhere **[13]**. It contains 974 genes clustered to nine groups with 47 features and 500 of the genes are used for testing. Clearly Table 5 (supplementary material) shows that the proposed method achieves an improvement of the same recognition accuracy with the sub-dimension based PNN (82.4%). As comparison, the normal PNN classification results are 79.6% accuracy. Similar results on the Annton dataset, containing 639 genes in five classes and 47 features, of which half are in the test set. As expected, the test set presents almost the same success as the sub-dimension based PNN, at 73% accuracy. The normal PNN could only obtain 283 correct out of 400 testing data. As shown in the tables (under supplementary material), the proposed method performs very closely to the sub-dimension based PNN which uses all sub-dimension features.

Conclusion:
Instead of considering all features of datasets in a classifier, our previous paper **[1]** implemented the PNN classification on single sub-dimensions. However, the number of combinations of sub-dimensions is large and this overall system computationally to complicated. In this paper, a feature evaluation and selection technique based on an entropy definition is used to measure the contribution of each sub-dimension. The sub-dimension with the lowest contribution to the overall classification is discarded. Experiments on two real world datasets and four microarray datasets show clearly that the achievement of the proposed technique is remarkable better than the normal PNN and as good as the sub-dimension based PNN. However the system complexity is significantly reduced and the classification speed is increased. The feature evaluation and selection are especially effective and convenient when the input features are large and the datasets are noisy. At the rank of the corresponding information gain ***G*,** the importance of the sub-dimension decreases while ***G*** reduces. Good performance selection occurs particularly at the top of the rank. However, how many sub-dimensions should be considered as important is a critical issue which needs to be investigated further.

References:
[1] Y. Wang *et al.*, *Lecture Notes in Engineering and Computer Science,* Hong Kong, 222 (2008)
[2] F. Specht, *Proc. IEEE Int. Conf. Neural Networks*, vol. 1, San Diego, CA, 525 (1988)
[3] F. Specht, *IEEE Trans. Neural Networks*, 111 (1990)
[4] F. Specht and P. D. Shapiro, *IJCNN-91-Seattle Int. Joint Conf. Neural Networks*, 1: 887 (1991)
[5] B. S. Y. Lam and H. Yan, *Physical Review E*, 74: 041906 (2006)
[6] E. Shannon, *The Bell System Technical Journal*, 27: 379 (1948)
[7] H. M. Lee *et al.*, *IEEE Trans. Syst., Man, Cybern. C*, 31 (2001)
[8] S. V. Kartalopoulos, *USA: IEEE Neural Networks Council*, 104-105 (1996)
[9] M. M. Lavalle *et al.*, *IEEE Proc.*, (2006)

[10] Z. Chi and H. Yan, 34: 12 (1995)
[11] R. Cho *et al.*, *Molecular Cell*, 2: 65 (1998)
[12] S. Chu *et al.*, *Journal of Bacteriology*, 188: 8178 (2006)
[13] M. L. Whitfield *et al.*, *Mol. Biol. Cell*, 13: 1977 (2002)
[14] http://www.ics.uci.edu/~mlearn.MLSymmary.html

Edited by P. Kangueane
Citation: Wang *et al.*, Bioinformation 3(3): 124-129 (2008)

Supplementary material

Classification method:
Probabilistic Neural Network:
Instead of using the conventional BPN, 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 **[1]**, **[2]**. 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:

$$\boxed{d(X)=\theta_A} \text{ if } \boxed{h_A l_A f_A(X) > h_B l_B f_B(X)}$$
$$\boxed{d(X)=\theta_B} \text{ if } \boxed{h_A l_A f_A(X) < h_B l_B f_B(X)} \qquad \rightarrow \qquad (1)$$

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 the 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 **[1]** and **[2]**, it is shown that a particular estimation of a probability density function of category θ_A is

$$\boxed{\begin{array}{l} f_A(X)=\dfrac{1}{(2\pi)^{p/2}\sigma^p}\dfrac{1}{m} \\ \cdot\displaystyle\sum_{i=1}^{m}\exp\left[-(X-X_{Ai})^t(X-X_{Ai})/(2\sigma^2)\right] \end{array}} \qquad \rightarrow \qquad (2)$$

where 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 to the Gaussian.

The PNN network consists of input units, two hidden layers, and output units. Figure 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, $\boxed{Z_i = X \cdot W_i}$, 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:

$$\boxed{\exp\left[(Z_i - 1)/\delta^2\right]}. \quad \rightarrow \quad (3)$$

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

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

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 such a 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 Figure 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 disadvantages of PNN **[8]**.

Sub-dimension:
The sub-dimension method **[4]** can be implemented by dividing the databases into smaller parts and applying the classification procedure to each part. Let x_{ij} be a matrix with i objects (rows) and j features (columns), and

$$\boxed{X = \left[A_1 \quad A_2 \cdots A_j \cdots A_d\right]}$$

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

$$\boxed{X = \left[B_1 \quad B_2 \cdots B_p\right]}$$
$$\boxed{B_j = \left[A_{j1} \quad A_{j2} \cdots A_{js}\right]}$$

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 .

Instead of considering all features as evidence for classification, the sub-dimension based PNN algorithm takes the sub-dimension B_j as the input pattern and implements the PNN classification to each sub-dimension respectively. The observable benefit of this approach is that results of each sub-dimension are hardly affected by features in other sub-dimensions. In a previous study, we simply employed the majority decision as the class label determination. We concluded 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:

$$\boxed{\begin{array}{l} Card\left(\left\{j : \left\|X_{1(B_j)} - X_{2(B_j)}\right\| < \left\|X_{1(B_j)} - X_{3(B_j)}\right\|\right\}\right) \\ > Card\left(\left\{j : \left\|X_{1(B_j)} - X_{2(B_j)}\right\| \ge \left\|X_{1(B_j)} - X_{3(B_j)}\right\|\right\}\right) \end{array}} \quad \rightarrow \quad (5)$$

where $Card(S)$ refers to the cardinality (or the number of elements) of the set S **[5]**. Applying Equation (5) to earlier experiments, we assigned object x_i to a group, if a majority of sub-dimensions x_{ij} are classified to that group.

Entropy Measure:
Theoretically, the initial entropy definition of a discrete variable X is as follows **[9]**:

$$\boxed{H_n = -\sum p_i \log_2 p_i}$$

where H_n represents the uncertainty or the entropy of the set of probabilities $p_1 \cdots p_n$.

In **[10]**, it is considered that objects are distributed into l classes. Let "w_1," "w_2," … "w_i," … "w_l," be the general information of each class. The information of the overall classification which adopts all features can be given as:

$$\boxed{I[C(p,l)] = -\sum_{i=1}^{l} P(w_i)\log_2 P(w_i)}, \quad \rightarrow \quad (6)$$

where $P(w_i)$ is the probability that one object falls in class w_i . It can be estimated from:

$$\boxed{P(w_i) = \frac{N(w_i)}{N}}, \quad \rightarrow \quad (7)$$

where $N(w_i)$ is the number of objects falling in class w_i, $i = 1,2,...,l$, and N is the total number of objects,

$$\boxed{N = \sum_{i=1}^{l} N(w_i)} \quad \rightarrow \quad (8)$$

Based on the definition above, the conditional probability can be used as the measure of the contribution of single sub-dimension entropy to the overall classification. For the j th input sub-dimension, the PNN classifier is applied. Data are classified into l clusters, R_{j1}, R_{j2}, … , R_{jk}, …, R_{jl}, $k = 1,2,...,l$. Then we estimate

$$\boxed{P(R_{jk}) = \frac{N(R_{jk})}{N}}, \quad \rightarrow \quad (9)$$

$$\boxed{P(w_i|R_{jk}) = \frac{N(w_i, R_{jk})}{N(R_{jk})}}. \quad \rightarrow \quad (10)$$

where $P(R_{jk})$ is the probability of objects falling in the k th cluster. The conditional probability $P(w_i|R_{jk})$ represents the probability of an object in cluster k to be classified into class w_i. $N(w_i, R_{jk})$ is the number of objects in cluster k that are classified into class w_i and $N(R_{jk})$ is the total number of objects in the k th cluster. Then we have

$$\boxed{N(R_{jk}) = \sum_{i=1}^{l} N(w_i, R_{jk})}. \quad \rightarrow \quad (11)$$

The j th sub-dimension entropy can be defined as:

$$\boxed{E(j) = -\sum_{k=1}^{K_j} P(R_{jk})\sum_{i=1}^{l} P(w_i|R_{jk})\log_2 P(w_i|R_{jk})}. \rightarrow \quad (12)$$

The corresponding information gain of the j th sub-dimension clustering is

$$\boxed{G(j) = I[C(p,l)] - E(j)}. \quad \rightarrow \quad (13)$$

Since the information of the overall classification $\boxed{I[C(p,l)]}$ is a constant for all sub-dimensions, a small $E(j)$ produces a large $G(j)$. If all objects within the k th cluster are from the same class w_i, and $w_i = k$, $E(j)$ reaches its minimum value 0, while $G(j)$ reaches its maximum value, $\boxed{I[C(p,l)]}$. Our classification results would therefore fully depend on a single sub-dimension. On the contrary, when $E(j)$ reaches its maximum value $\log_2 L$, $G(j)$ will reach its minimum value, $\boxed{I[C(p,l)] - \log_2 L}$, where $\boxed{P(w_i | R_{jk}) = 1/l}$, $i = 1,2,...,l$ and $\boxed{k = 1,2,...,l}$. Our classifications in the sub-dimension would therefore perform at poor accuracy. To sum up, $G(j)$ represents the contribution that the j th input sub-dimension makes to the general classification. The larger $G(j)$ is, the greater the sub-dimension contribution is **[10]**.

Applying the entropy measure to the proposed method, we can choose sub-dimensions with large $G(j)$ values and abandon the smaller ones to reduce the feature inputs. For each sub-dimension, we compute the entropy and calculate the corresponding information gain G. The process is carried out on all sub-dimensions. A set of G is attained which represent the donations of each sub-dimensions. After the comparison of the G value, the sub-dimension with the lowest G is discarded. Then we vote for each testing object and choose the majority result as the class label.