Berrar D., Downes C.S., and Dubitzky W. (2003) A probabilistic neural network for gene selection and classification of microarray data. Proc. Intl. Conference on Artificial Intelligence (IC-AI03), Las Vegas, USA, pp. 342-349.

A Probabilistic Neural Network for Gene Selection and Classification of Microarray Data

Daniel Berrar, C. Stephen Downes, Werner Dubitzky
School of Biomedical Sciences,
University of Ulster at Coleraine,
BT52 1SA, Northern Ireland

Abstract

In this paper, we present the mathematical foundations of a probabilistic neural network for gene selection and classification of high-dimensional microarray data. We present a catalogue of features that a classification system for microarray data should incorporate. We then use this catalogue and compare the theoretical properties of probabilistic neural networks with support vector machines with regard to their suitability for multiclass cancer prediction. We compare the classification performance of a probabilistic neural network with the performance of a support vector machine on a multiclass microarray data set. The results of the theoretical and practical comparison suggest that the probabilistic neural network approach is to be preferred over support vector machines for multiclass cancer classification using microarray data.

Keywords: probabilistic neural network, support vector machine, microarray, cancer classification.

1. Introduction

Traditionally, the classification of complex diseases such as cancer has been performed on the basis of non-molecular criteria such as tumor tissue type, pathological features, and clinical stage. In spite of recent medical progress, the diagnosis of cancer remains a challenging and very complex task. Existing cancer classes can be heterogeneous and may comprise diseases that are molecularly distinct and follow different clinical courses. Microarray technology allows the monitoring of gene expression levels in cells for thousands of genes simultaneously, and may lead to a more complete understanding of the molecular variations among tumors and hence to a finer and more reliable classification. The reliable and precise classification of samples is essential for diagnosis, therapy, and prediction of cancer. This explains why the scientific task of classification has received considerable attention in microarray analysis. Microarray analysis of differential gene expression has been used to distinguish between different subtypes of cancer such as lung adenocarcinoma [1], colorectal neoplasm [2] and breast cancer [3], and to predict clinical outcomes in breast cancer [4,5] and lymphoma [6]. Many statistical and machine learning methods have been proposed to address the task of tumor sample classification, e.g., test statistics [7], neural networks approaches [2,8], and decision trees [9]. Most machine-learning classifiers are generally designed to operate on a large number of samples over relatively few variables. Therefore, high-dimensional microarray data present a major challenge for these classifiers. A number of recent publications report on the successful application of support vector machines (SVMs) to the classification of high-dimensional microarray data [10,11,12]. SVMs are based on sound statistical principles, and also have demonstrated their excellent classification performance on a wide range of complex problems in bioinformatics. To date, SVMs are considered a state-of-the-art classifier for microarray data [13]. However, the method of choice for classifying microarray data is not available, but it depends on the data properties and the type of the classification problem (e.g., binary vs. multiclass prediction), according to the *No-Free-Lunch* theorem [14]. Consequently, the

general question "*Which classifier is the method of choice for microarray data analysis?*" is ill-posed. There are few comparative studies on the suitability of various methods for classifying microarray data [15].

In this paper, we propose to assess the suitability of a classifier on the basis of qualitative and quantitative criteria. Qualitative criteria refer to general features of the classifier, whereas quantitative criteria refer to the classification performance on concrete data sets. The outline of this paper is as follows: Firstly, we present a catalogue of requirements that a classifier for cancer gene expression data should meet. Then, we show that SVMs do not fulfill all these requirements. Thirdly, we present the model of a probabilistic neural network (PNN) that is able to address all but one of the qualitative criteria. Finally, we compare the classification performance on a microarray data of multiple cancer types. There exist >100 types of cancer and even more subtypes, so that practical applications require multiclass methodologies for molecular classification [16].

2. Requirement Analysis for Molecular Classifiers

Table 1 summarizes the (minimal) set of properties that a molecular classifier should have.

Table 1: Minimal set of properties for molecular classifiers.

#	Property	Explanation / Remark
1	Low variance and low bias	The classifier is characterized by a low variance and a low bias; this is a prerequisite for a good generalization ability.
2	Handling of missing values	The model is able to handle missing values adequately.
3	Addressing the *large-p-small-n* problem	Also known as curse of dimensionality, this due to an imbalance of the number variables (p) and the number of observations (n).
4	Handling of skewed class distributions	Class skewness refers to any imbalance in the class distributions, e.g., one class contains only very few cases, while another class contains very many cases.
5	Providing robust solutions	The model's output is deterministic, and slight changes to the learning data do not affect the model's performance. The model is able to cope with noisy data.
6	Incorporation of asymmetrical misclassification costs	The costs that are associated with misclassifications depend on the class membership of the misclassified cases. The costs associated with false positive and false negative classifications are context-dependent.
7	Detection of outliers	(self-explanatory)
8	Identification of important variables	The model is able to identify the variables that are most important for the classification task.
9	Handling of multiclass problems	The model is able to solve classification tasks involving more than two target classes.
10	Quantification of uncertainty and confidences	The model is able to quantify the uncertainty associated with its decisions, for example, state confidence level in terms of Bayesian confidences.
11	Handling of mixed variables	The model is able to handle data expressed as mixture of ordinal, categorical, and numeric variables.
12	Easy-to-use	The model is easy to understand and use (parameter settings, output interpretation).
13	Incorporation of domain knowledge	The model is able to make use of existing domain knowledge, rather than purely relying on the data at hand.

Despite the excellent credentials of SVMs, these models are not able to address all the issues that are involved in the classification task of microarray data. SVMs meet the requirements (1) – (9) outlined in Table 1. Concerning (9), however, it should be noted that SVMs are in principle binary classifiers and unable to solve problems that involve more than two classes. To solve multiclass problems, we have to combine the binary classifiers by using e.g., *one-versus-all* (OVA), *all-pairs*, or *hierarchical partitioning* approaches [16]. The optimal design for combining binary classifiers is still an open issue; however, recent studies report that the OVA approach provides the best results for combining SVMs [10,12,16]. In the OVA approach, we build one SVM for each single class. Each SVM is trained to separate between one class and all other classes. We can combine the SVMs in a directed acyclic graph (DAG) to one single model [17]. For a problem involving k

classes, $k-1$ decision nodes will be evaluated to classify a new object.

Concerning (10): Although some work has been done to add probabilistic semantics to the output of SVMs [18], the interpretation of their output remains difficult. SVMs are not able to provide estimated conditional probabilities for their classifications. It has been suggested to interpret the distance between a case and the decision hyperplane as a measure of the model's confidence: the larger the margin, the more confident the SVM [10,16]. However, this measure relies on a geometrical interpretation and can be misleading, because it depends on the distance metric being used and does not allow any interpretation in terms of estimated class posteriors.

Concerning (11): Gene expression data are often represented as discrete values, where 1 indicates over-expression, 0 indicates normal expression, and –1 indicates under-expression. Discrete data, however, present a problem for SVMs and require adequate rescaling [19].

Concerning (12): SVMs are characterized by the choice of the kernel function, which has a major effect on the classification performance. Furthermore, the choice of the regularization parameter and the window width for the Gaussian kernel have an influence on the classification performance. However, finding the optimal design and the optimal parameter settings involve a multivariate optimization problem, which is a non-trivial task for a user who is not familiar with SVMs.

Concerning (13): SVMs are statistical classifiers that rely only on the data set at hand. Apart from manual variable selection, the user has no means to incorporate domain knowledge into the SVM model.

3. Probabilistic Neural Networks

PNNs are based on Bayes' decision strategy and Parzen's method of density estimation. D. Specht has proposed a four-layered feed-forward network topology to implement PNNs [20]. He splits the algorithm of the Bayes-Parzen classifier into a number of simple processes, which can be computed in parallel. PNNs have shown excellent performance in classifying microarray data [21]. An important parameter in these models is the width of the Parzen window, which is also referred to as *smoothing factor* σ. We propose the following framework for building a PNN classifier: (1) Optimize the smoothing factor globally using a homoscedastic PNN; (2) Initialize local smoothing factors for each case of the learning set using the global smoothing factor, and then optimize these local factors gradually (heteroscedastic PNN).

3.1 Homoscedastic PNN

Let the prior probability that a sample $\vec{x}$ belongs to population k be denoted as h_k. The costs associated with a misclassification of a sample belonging to population k is denoted as c_k. The estimated conditional probability that a specific sample belongs to class k, $\hat{p}(k \mid \vec{x})$, is given by the probability density function $\hat{f}_k(\vec{x})$. Then an unknown sample $\vec{x}$ is classified into class i if

$$h_i \cdot c_i \cdot \hat{f}_i(\vec{x}) > h_j \cdot c_j \cdot \hat{f}_j(\vec{x}) \qquad (1)$$

for all classes $j \neq i$ (*Bayes' decision criterion*). The estimated density for the j^{th} class at the new case $\vec{x} = (x_1, x_2, ...x_p)$ is given by Equation 2*:*

$$\hat{f}_j\left(x_1, x_2, ...x_p\right) = \frac{1}{\left(\sqrt{2\pi}\right)^p \sigma_1\sigma_2,...,\sigma_p m_j} \cdot \sum_{i=1}^{m_j} W\left(\frac{d\left(x_1, x_{1,ij}\right)}{\sigma_1}, \frac{d\left(x_2, x_{2,ij}\right)}{\sigma_2}, ..., \frac{d\left(x_p, x_{p,ij}\right)}{\sigma_p}\right) \qquad (2)$$

In Equation 2, m_j indicates the number of training cases in the j^{th} class; σ_k indicates the smoothing factor for the k^{th} element in $\vec{x}$. The component $x_{1,ij}$ is the first element of the i^{th} case of class j. Furthermore, $d(x_1, x_{1,ij})$ indicates the distance between the first element of the test case and $x_{1,ij}$. W indicates the *multivariate kernel* or *weighting function*. If the estimated density is used for classification purposes only, then the constant factor $1/((\sqrt{2\pi})^p \sigma_1\sigma_2,...,\sigma_p)$ in Equation 2 can be neglected. It should be noted that Equation 2 allows the handling of missing values without explicit imputation methods. For example, if either x_1 or $x_{1,ij}$ is missing, then the kernel function will not include the distance $d(x_1, x_{1,ij})$ (cf. Property 2 of Table 1). Furthermore, Equation 2 allows the use of feature-dependent distance metrics, i.e. $d_1, d_2,..., d_p$ for p different

variables, so that the PNN is able to cope with mixed variables (cf. Property 11 of Table 1).

In practical applications, the most commonly used kernel is the unweighted Gaussian as shown in Equation 3:

$$W(d) = e^{-d^2} \tag{3}$$

where $W(w_1, w_2, ..., w_n) = \prod_i^n W(w_i)$. We obtain the simplified multiple of a density function for class *j* as shown in Equation 4:

$$\hat{g}_j(\vec{x}) = \frac{1}{m_j} \sum_{i=1}^{m_j} e^{-d(\vec{x}, \vec{x}_{ij})} \tag{4}$$

In the following sections, we use the Euclidean distance, i.e.: $d(\vec{x}, \vec{x}_{ij}) = \sum_{k=1}^{p} \left(\frac{x_k - x_{k,ij}}{\sigma_j} \right)^2$. (We do not consider the square root here and use exp(– *d*) in the following sections.) The following diagram illustrates the Bayes' decision strategy that is implemented in the basic PNN.

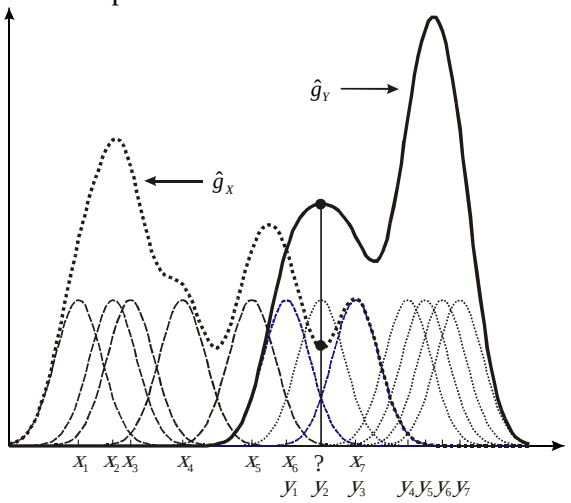

Figure 1: Bayes' decision strategy for a two-class problem. The dotted curve ($\hat{g}_X$) is the sum of the dotted Gaussian "bumps" centered at the cases of class *X*. The solid curve ($\hat{g}_Y$) is the sum of the dotted "bumps" centered at the cases of class *Y*. The question mark represents the unknown test case.

In the simplified univariate example depicted in Figure 1, we have two classes, *X* and *Y*, each comprising seven cases: $X = \{x_1, x_2, ..., x_7\}$ and $Y = \{y_1, y_2, ..., y_7\}$. Around each case, a bell curve is constructed with a constant smoothing factor σ. In this example, we have equal class priors (i.e. each class has the same prior probability of occurring). We assume equal costs for misclassification, so that the Bayes' decision criterion is based on the estimated (simplified) class densities alone. In the example, the PNN classifies the unknown test case, represented by the question mark, as a member of class *Y*, because $\hat{g}_Y(?) > \hat{g}_X(?)$. Figure 2 shows the PNN implementation for the example data of Figure 1. Each neuron in the pattern layer receives as input the unknown test case. In the next step, each pattern neuron N_i computes the distance between the test case and the training case that it "harbors". The distance *d* is then weighted by a global σ. The activation of neuron N_i is given by exp(– *d*); the pattern neuron N_i feeds this activation to the summation neuron. The summation neurons sum up the activations of the associated pattern neurons and feed their result into the output neuron, which outputs the estimated conditional posteriors.

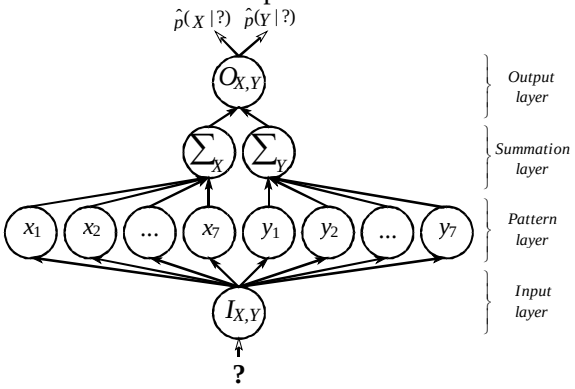

Figure 2: PNN implementation for the example data of Figure 1. The PNN contains 1 input neuron (for the 1 test case), 14 pattern neurons (7 for each class), 2 summation neurons (1 for each class), and 1 output neuron.

By allowing each variable to have its individual smoothing factor, we can extent the PNN to a feature selector: the smaller σ_i, the more important the i^{th} variable, and vice versa. The following section explains the motivation for this advanced model and the mathematical details.

3.2 Heteroscedastic PNN

Figure 3 shows an example of two overlapping classes, each containing uniformly distributed two-dimensional points.

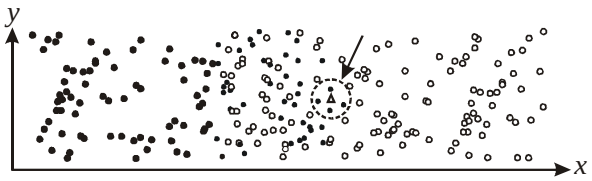

Figure 3: Two overlapping classes, each containing uniformly distributed points. The black points belong to class *A*, and the white points belong to class *B*. The encircled triangle is the unknown test point.

Although the unknown test case is surrounded by cases of class *A*, the test case is obviously more likely to be a member of class *B*. Clearly, the decision is based on the *x*-variable only; the *y*-variable is irrelevant for the classification. Suppose that we choose the same smoothing factor σ for both variables. A small value for σ will result in magnifying the distances in Equation 2. The closer a training point to the test point, the greater the value of the kernel function. Consequently, the PNN will base its decision only on the nearest neighbors of the test case, which will result in a misclassification in the scenario depicted in Figure 3. A large value of σ will result in relatively large values of the kernel function, so that the PNN will consider not only the points in the immediate neighborhood of the test point, but also the points that are farther away. However, the distance between the test point and any training point has two components: the distance between the points based on the *x*-variable *and* the *y*-variable. Suppose that in the example data in Figure 3 the variance of the *x*-values is much lower than the variance of the *y*-values. Then the distance will be dominated by the *y*-variable, which has no discriminatory information for the classifier. Consequently, the smoothing factor for the *x*-variable, σ_x, should be smaller than the smoothing factor for the *y*-variable, σ_y, to emphasize the importance of the *x*-variable. As a function of the classification performance in the training phase, the individual smoothing factors have to be adapted.

To assess the classification performance of the PNN, we should take into account the model's confidence in its decisions. Clearly, a misclassification with high confidence is more severe than a misclassification with low confidence. On the other hand, a correct classification with high confidence is preferable to a correct classification with low confidence. Suppose that the number of test cases is *t*. Then the mean squared error (*MSE*) is given by Equation 5:

$$MSE = \frac{1}{t}\sum_{i=1}^{t}\left[\begin{array}{l}\left(1-\hat{p}\left(c_k \mid \vec{x}_i \in c_k\right)\right)^2 \\ +\sum_{j\neq k}^{K}\hat{p}\left(c_j \mid \vec{x}_i \in c_k\right)^2\end{array}\right] \qquad (5)$$

In total, we have *K* classes. Suppose that the i^{th} test case, $\vec{x}_i$, is a member of class c_k. Then ideally $\hat{p}(c_k \mid \vec{x}_i \in c_k) = 1$. If, e.g., $\hat{p}(c_k \mid \vec{x}_i \in c_k) = 0.8$, then the error for this classification is $(1 - 0.8)^2 = 0.04$. The second term takes into account the estimated probabilities for remaining (incorrect) classes. Furthermore, this terms penalizes an unequal distribution of the estimated class posteriors. The following example illustrates this idea. Consider the following two classification scenarios for 4 classes *A*, *B*, *C*, and *D*, and let the test case be a member of class *A*.

Table 2:Unequal distribution of estimated class posteriors.

	Confidence				
Class	*A*	*B*	*C*	*D*	*MSE*
Scenario 1	0.7	0.1	0.1	0.1	0.12
Scenario 2	0.7	0.0	0.0	0.3	0.18

In scenario 1, the probabilities for the wrong classes are equally distributed, whereas in scenario 2, the probability is concentrated in class *D*, which results in a larger *MSE*. Clearly, a more balanced distribution of the probabilities for the incorrect classes is preferable, because an imbalanced distribution is more likely to lead to a misclassification.

The *MSE* provides a continuous error criterion that we can use for optimizing the smoothing factors. Let $sse(\vec{x}_i)$ be the sum squared error for the i^{th} training case, and let the actual class of this case be c_k. Then the derivative of $sse(\vec{x}_i)$ with respect to the smoothing factor for the j^{th} variable, σ_j, is given by Equation 6:

$$\frac{\partial sse(\vec{x}_i)}{\partial \sigma_j} = 2\left(\hat{p}(c_k \mid \vec{x}_i \in c_k) - 1\right)\cdot\frac{\partial \hat{p}(c_k \mid \vec{x}_i \in c_k)}{\partial \sigma_j} + 2\sum_{l\neq k}^{K}\left(\hat{p}(c_l \mid \vec{x}_i \in c_k)\cdot\frac{\partial \hat{p}(c_l \mid \vec{x}_i \in c_k)}{\partial \sigma_j}\right) \qquad (6)$$

Equation 6 leads to the gradient

$$\nabla E = \left(\frac{\partial sse(\vec{x}_i)}{\partial \sigma_1}, \frac{\partial sse(\vec{x}_i)}{\partial \sigma_2}, ..., \frac{\partial sse(\vec{x}_i)}{\partial \sigma_p} \right)$$

for the i^{th} test case. To derive the gradient of *MSE*, we calculate the gradient for all test cases and summarize the results; the normalizing factor t^{-1} in Equation 5 can be neglected. The resulting gradient is given in Equation 7:

$$\nabla E_{MSE} = \left(\sum_{i=1}^{t} \frac{\partial sse_i(\vec{x}_i)}{\partial \sigma_1}, ..., \sum_{i=1}^{t} \frac{\partial sse_i(\vec{x}_i)}{\partial \sigma_p} \right) \quad (7)$$

Finding the minimum of the error functional given by *MSE* is a multivariate optimization problem. To solve this problem, we need the derivatives of the estimated posteriors with respect to the smoothing factors. The equations in this section follow the notation by T. Masters [22]. The estimated probability that a test case $\vec{x}_i$ is a member of class c_k is given by Equation 8:

$$\hat{p}\left(c_k \mid \vec{x}_i \in c_k\right) = \frac{m_k^{-1} \cdot \sum_{k=1}^{m_k} e^{-d(\vec{x}_i, \vec{x}_k)}}{n^{-1} \cdot \sum_{j=1}^{n} e^{-d(\vec{x}_i, \vec{x}_j)}} := \frac{h_k}{s} \quad (8)$$

The nominator is the sum of the kernel function values for the test case and the cases of the training set that belong to class c_k. The number of cases of class c_k is m_k. The denominator is the sum of the kernel function values for the test case and all cases of the training set, regardless of their class membership. The total number of training cases is n. The first derivative of the nominator is given in Equation 9:

$$\frac{\partial h_k}{\partial \sigma_j} = 2 m_k^{-1} \cdot \sum_{k=1}^{m_k} \left[e^{-d(\vec{x}_i, \vec{x}_k)} \cdot \frac{\left(x_{ij} - x_{k,ij}\right)^2}{\sigma_j^{\,3}} \right] := a_{kj} \quad (9)$$

The first derivative of the denominator is given by Equation 10:

$$\frac{\partial s}{\partial \sigma_j} = 2 n^{-1} \cdot \sum_{k=1}^{n} \left[e^{-d(\vec{x}_i, \vec{x}_k)} \cdot \frac{\left(x_{ij} - x_{k,ij}\right)^2}{\sigma_j^{\,3}} \right] := b_j \quad (10)$$

Equations 9 and 10 lead to the derivative of the estimated probability as shown in Equation 11.

$$\frac{\partial \hat{p}\left(c_k \mid \vec{x}_i \in c_k\right)}{\partial \sigma_j} = \frac{a_{kj} - \hat{p}\left(c_k \mid \vec{x}_i \in c_k\right) \cdot b_j}{s} \quad (11)$$

To find a starting point that is close to the global minimum, we can apply the homoscedastic PNN, and determine the optimal value for the global σ in a cross-validation procedure. Then, we initialize the σ_j of the advanced model using this σ. Using the gradient of Equation 7, we find an optimal set of smoothing factors $\sigma_1, \sigma_2, \ldots, \sigma_p$, which addresses two issues:

1. different smoothing factors minimize the error criterion of Equation 5;
2. the value of σ_j can be interpreted as the importance of the j^{th} variable.

Thereby, the PNN integrates feature selection and classification within a single and consistent framework.

4. Comparative Study

4.1 Material

To compare the classification performance of PNNs and SVMs, we choose a multiclass cancer microarray data set, the NCI60 data set [23]. The NCI60 data set comprises gene expression profiles of 60 cell lines. The data set includes nine different cancer classes: central nervous system (*CNS*, 6 cases), breast (*BR*, 8 cases), renal (*RE*, 8 cases), lung (*LC*, 9 cases), melanoma (*ME*, 8 cases), prostate (*PR*, 2 cases), ovarian (*OV*, 6 cases), colorectal (*CO*, 7 cases), and leukemia (*LE*, 6 cases). The gene expression data comprise mainly ESTs of known and unknown function given by the negative logarithm of the ratio between the red and green fluorescence of the signals. After data preprocessing, the microarray matrix contains 1,407 genes. We do not apply any feature selection method to reduce the number of irrelevant or redundant genes, because we are interested in the performance of the models in the presence of noise (cf. Property 5 in Table 1).

4.2 Methods

We choose the OVA approach [10,12,16] and combine the binary classifiers using the DAG [17]. We choose a simple dot product kernel for the SVM; Furey et al. have demonstrated that this simple kernel performs better than radial and polynomial kernels for a comparable data set [24]. We use the sequential minimal optimization algorithm to train the SVM [25]. To allow for a fair comparison of the methods, we apply only the homoscedastic PNN that does not implement the implicit feature selection. Furthermore, we

choose the dot product kernel for the PNN as well. To optimize the smoothing factor σ, we take into account the continuous error criterion of Equation 5.

Given the limited size of the NCI60 data set, we evaluate the classification performance using the leave-one-out cross-validation (LOOCV) procedure: remove a single sample from the data set, use the remaining 59 samples as training set, and test the model's ability to classify the hold-out sample; iterate this procedure until each sample was used as hold-out case. Table 3 shows the resulting confusion matrix.

Table 3: Confusion matrix for LOOCV classification results. (The results of the PNN are shown in bold.)

	Real class								
	CNS	*BR*	*RE*	*LC*	*ME*	*PR*	*OV*	*CO*	*LE*
CNS	**6**|5	**1**|1	-|-	-|1	-|-	-|-	-|-	-|-	-|-
BR	-|1	**5**|4	**1**|1	**1**|2	-|-	-|-	**1**|1	-|-	-|-
RE	-|-	-|-	**7**|7	**2**|1	-|-	**1**|2	-|-	-|-	-|-
LC	-|-	**1**|1	-|-	**5**|5	-|1	-|-	**1**|1	-|-	-|-
ME	-|-	-|-	-|-	-|-	**7**|7	-|-	-|-	-|-	-|-
PR	-|-	-|-	-|-	-|-	**1**|-	**1**|-	-|-	-|-	-|-
OV	-|-	-|1	-|-	-|-	-|-	-|-	**4**|4	-|-	-|-
CO	-|-	-|1	-|-	**1**|-	-|-	-|-	-|-	**7**|7	-|-
LE	-|-	-|-	-|-	-|-	-|-	-|-	-|-	-|-	**6**|6

The PNN classified 48 of 60 cases correctly, while the SVM classified 45 cases correctly.

5. Discussion

The Ugly Duckling Theorem states that there exists no problem-independent set of optimal features, but it is the type of the task, the problem at hand, and the optimization criterion that determine which features are "best" [26]. In this paper, we have demonstrated how the homoscedastic PNN can be extended to feature selector by allowing an individual smoothing factor for each variable. The optimization criterion respects the model's confidences, and the smoothing factors are optimized in such a way that the mean squared error of the Bayesian confidences is minimized. This approach provides a higher "reward" for correct classifications with high confidence than for correct classifications with low confidence, and a higher "penalty" for misclassifications with high confidence than for misclassifications with low confidence. By varying the smoothing factors, the PNN assigns a higher weight to those variables that help in minimizing the error functional. Thereby, the PNN strongly couples implicit feature selection with classification performance, which itself is defined in terms of confidence. This approach is fundamentally different to other classifiers that require explicit dimension reduction procedures.

In contrast to SVMs, PNNs can inherently cope with multiclass problems (property 9 in Table 1); unlike SVM s, PNNs do not need to be combined. Most importantly, PNNs provide Bayesian confidences for their decisions (property 10 in Table 1). Future directions of molecular classification of cancer will probably focus on integrating microarray data with clinical data [13]. Unlike SVMs, PNNs are able to cope easily with mixed variables. PNNs might deploy their full potential in the light of microarray data enriched by clinical data. However, the incorporation of domain-specific knowledge into the models (both SVMs and PNNs) is still an open issue (property 13 of Table 1).

In the present study, the PNN outperformed the SVM with respect to classification accuracy. However, further studies will be necessary to confirm whether the theoretical credentials of PNN will transfer to practical performance.

Our future research will focus on a broad comparative study on the performance of various machine learning techniques for classifying different cancer microarray data sets.

6. References

[1] Bhattacharjee A., Richards W.G., Staunton J., Li C., Monti S., Vasa P., Ladd C., Beheshti J., Bueno R., Gillette M., Loda M., Weber G., Mark E.J., Lander E.S., Wong W., Johnson B.E., Golub T.R., Sugarbaker D.J., Meyerson M., Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses. *Proc. Natl. Acad. Sci. USA* **98**(24):13790-13795, (2001).

[2] Selaru F.M., Xu Y., Yin J., Zou T., Liu T.C., Mori Y., Abraham J.M., Sato F., Wang S., Twigg C., Olaru A., Shustova V., Leytin A., Hytiroglou P., Shibata D., Harpaz N., Meltzer S.J., Artificial neural networks distinguish among subtypes of neoplastic colorectal lesions. *Gastroenterology* **122**:606-613, (2002).

[3] Hedenfalk I., Ringnér M., Ben-Dor A., Yakhini Z., Chen Y., Chebil G., Ach R., Loman N., Olsson H., Meltzer P., Borg A., Trent J.: Molecular classification of familial non-BRCA1/BRCA2 breast cancer. *Proc. Natl. Acad. Sci. USA* **100**(5):2532-2537 (2003).

[4] van't Veer L.J., Dai H.Y., van de Vijver M.J., He Y.D.D., Hart A.A.M., Mao M., Peterse H.L., van der Kooy K., Marton M.J., Witteveen A.T., Schreiber G.J., Kerkhoven R.M., Roberts C., Linsley P.S., Bernards

R., Friend S.H., Gene expression profiling predicts clinical outcome of breast cancer. *Nature* **415**:530-536, (2002).

[5] West M., Blanchette C., Dressman H., Huang E., Ishida S., Spang R., Zuzan H., Olson J.A., Marks J.R., Nevins J.R., Predicting the clinical status of human breast cancer by using gene expression profiles. *Proc. Natl. Acad. Sci. USA* **98**(20):11462-11467, (2001).

[6] Shipp M.A., Ross K.N., Tamayo P., Weng A.P., Kutok J.L., Aguiar R.C.T., Gaasenbeek M., Angelo M., Reich M., Pinkus G.S., Ray T.S., Koval M.A., Last K.W., Norton A., Lister T.A., Mesirov J., Neuberg D.S., Lander E.S., Aster J.C., Golub T.R., Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning. *Nature Medicine* **8**:68-74, (2002).

[7] Jaeger J., Sengupta R., Ruzzo W.L., Improved gene selection for classification of microarrays, Proceedings of the Pacific Symposium on Biocomputing 8, World Scientific, New Jersey/London/Singapore/Hong Kong, pp. 17-29, (2003).

[8] Khan J., Wei J.S., Ringnér M., Saal L.H., Ladanyi M., Westermann F., Berthold F., Schwab M., Antonescu C.R., Peterson C., Meltzer P.S., Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine* **7**(6):673-679, (2001).

[9] Berrar D., Granzow M., Dubitzky W., Stilgenbauer S., Wilgenbus, K. D. H., Lichter P., Eils R., New Insights in Clinical Impact of Molecular Genetic Data by Knowledge-driven Data Mining. In Proc. 2nd Int'l Conference on Systems Biology, Omnipress, pp. 275-281, (2001).

[10] Ramaswamy S., Tamayo P., Rifkin R., Mukherjee S., Yeang C.H., Angelo M. Ladd C., Reich M., Latulippe E., Mesirov J.P., Poggio T., Gerald W., Loda M., Lander E.S., Golub T.R., Multiclass cancer diagnosis using tumor gene expression signatures, *Proc. Natl. Acad. Sci. USA.* **98**(26):15149-15154, (2001).

[11] Brown M.P.S., Grundy W., Lin D., Cristianini N., Sugnet C., Furey T., Ares M., Jr., Haussler D., Knowledge-based analysis of microarray gene expression data by using support vector machines, *Proc. Natl. Acad. Sci. USA.* **97**(1):263-267, (2000).

[12] Mukherjee S., Classifying microarray data using support vector machines, in Berrar D., Dubitzky W., Granzow M. (eds.), *A Practical Approach to Microarray Data Analysis*, Kluwer Academic Publishers, Boston, pp. 166-185, (2002).

[13] Slonim D.K.: From patterns to pathways: gene expression data analysis comes of age, The Chipping Forecast II, *Nature Genetics*, Supplement, pp. 502-508, (2002).

[14] Wolpert D., Macready W.: No Free Lunch Theorems for Optimization, IEEE Transactions on Evolutionary Computation (**1**)1, pp. 67-82, (1997)

[15] Dudoit S., Fridlyand J., Speed T.P.: Comparison of discrimination methods for the classification of tumors using gene expression data. Technical Report 576, Department of Statistics, University of California at Berkeley, Berkeley, CA, (2000).

[16] Yeang C.H., Ramaswamy S., Tamayo P., Mukherjee S., Rifkin R.M., Angelo M., Reich M., LanderE., Mesirov J., Golub T.: Molecular classification of multiple tunor types, Bioinformatics **17**(1), pp. S316-S322, (2001).

[17] Platt J., Christianini N., Shawe_Taylor J., Large margin DAGs for multiclass classification, In Advances in Neural Information Processing Systems 12, S. A. Solla, T. K. Leen, K.-R. Mueller, eds., Cambridge, MA, MIT Press, (2000).

[18] Mukherjee S., Rifkin R., Support vector machines classification of microarray data, MIT Artificial Intelligence Laboratory Research Abstracts, http://www.ai.mit.edu/research/abstracts/abstracts2001/machine-learning/machine-learning.shtml, (2001).

[19] Burges C.: A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery **2**, p. 159, (1998).

[20] Specht D.F., Probabilistic Neural Networks. *Neural Networks*, Vol. 3, pp. 109-118, (1990).

[21] Berrar D., Downes C.S., Dubitzky W., Multiclass cancer classification using gene expression profiling and probabilistic neural networks, Proceedings of the Pacific Symposium on Biocomputing 8, World Scientific, New Jersey/London/Singapore/Hong Kong, pp. 5-16, (2003).

[22] Masters T.: *Advanced Algorithms for Neural Networks – A C++ Sourcebook*. John Wiley & Sons, Academic Press, pp. 201-205, (1995).

[23] Scherf U., Ross D., Waltham M., Smith L., Lee J., Tanabe L., Kohn K., Reinhold W., Myers T., Andrews D., Scudiero D., Eisen M., Sausville E., Pommier Y., Botstein D., Brown P., Weinstein J., A gene expression database for the molecular pharmacology of cancer. *Nature Genetics* **24**(3):236-244, (2000).

[24] Furey T.S., Christianini N., Duffy N., Bednarski D.W :, Schummer M., Haussler D.: Support vector machine classification and validation of cancer tissue samples using microarray expression data, Bioinformatics **16**(10), pp. 906-914, (2000).

[25] Platt J.: Sequential minimal optimization: A fast algorithm fro training support vector machines, in Schoelkopf B., Burges J.C., Smola A.J. (eds.), *Advances in Kernel Methods – Support Vector Learning*, Cambridge, MA, MIT Press, pp. 185-208, (1999).

[26] Duda R.O., Hart P.E., Stork D.G.: *Pattern Classification*, 2nd edition, John Wiley & Sons, New York, p. 461, (2000).