

The LSA Algorithm Applied to Gene-Expression Data Analysis*

Andreas Albrecht¹, Staal A. Vinterbo², and Lucila Ohno-Machado^{2,3}

¹ Univ. of Hertfordshire, Computer Science Dept., Hatfield, Herts AL10 9AB, UK
A.Albrecht@herts.ac.uk

² Harvard Medical School, Decision Systems Group, 75 Francis Str., Boston, MA, USA
{staal,machado}@dsg.harvard.edu

³ MIT, Division of Health Sciences and Technology, Cambridge, MA, USA

Abstract. We investigate the use of perceptrons for classification of microarray data where we use two datasets that were published in Khan et al. [17] and Golub et al. [13]. The classification problem studied by Khan et al. is related to the diagnosis of small round blue cell tumours of childhood (SRBCT) which are difficult to classify both clinically and via routine histology. Golub et al. study acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL), where currently no single test is available to establish the differential diagnosis. We used a simulated annealing-based method in learning a system of perceptrons, each obtained by resampling of the training set. Our results are comparable to those of Khan et al. and Golub et al., indicating that there is a role for perceptrons in the classification of tumours based on gene expression data. We also show that it is critical to perform feature selection in this type of models, i.e., we propose a method for identifying genes that might be significant for the particular tumour types. For SRBCTs, zero error on test data has been obtained for 10 out of 2308 genes only; for the ALL/AML problem, the best results are for about 50 genes out of a total number of 7129 inputs. Furthermore, we provide evidence that Epicurean-style learning is essential for obtaining the best classification results.

1 Introduction

Measuring gene expression levels is important for understanding the genetic basis of diseases. The simultaneous measurement of gene expression levels for thousands of genes is now possible due to microarray technology [20, 21]. Data derived from microarrays are difficult to analyze without the help of computers, as keeping track of thousands of measurements and their relationships is overwhelmingly complicated. Several authors have utilized unsupervised learning algorithms to cluster gene expression data [10]. In those applications, the goal is to find genes that have correlated patterns of expression, in order to facilitate the discovery of regulatory networks. Recent publications have begun to

*Research partially supported by EPSRC Grant GR/R72938/01 and by the Taplin award from the Harvard/MIT Health Sciences and Technology Division.

deal with supervised classification for gene expression derived from microarrays [11]. The goal in these applications is usually to classify cases into diagnostic or prognostic categories that are verifiable by a "gold-standard". Additionally, researchers try to determine which genes are most significantly related to the category of interest. Since the number of measurements is very large compared to the number of arrays, there is tremendous potential for overfitting in models that do not utilize a pre-processing step for feature selection. The feature selection process itself is of interest, as it helps to determine the relative importance of a gene in the classification. Approaches for feature selection in the context of gene expression analysis are currently being investigated. Developing a strategy for selecting genes that are important in a classification model, regardless of their absolute expression levels, is important in this context. Determining whether simple learning models can be successfully applied to this type of problem is also important. In this paper, we propose an algorithm for learning perceptrons based on simulated annealing and a resampling strategy and show that it can be successfully applied to the analysis of gene-expression data. The basic algorithm has been introduced in [7] and is called LSA algorithm. One of the key features of our approach is training perceptrons [19, 22] on randomly selected subsets of the entire sample set. In the present application, it turns out that the learning procedure is finished in almost all cases with zero error, which might be caused by the small amount of samples available (although the size of each sample is quite large, e.g., 7129 gene data in the ALL/AML problem). However, even in the case of image classification considered in [5, 6], the same effect has been observed on subsets of about 160 samples, each of them with 14161 inputs. Thus, in almost all cases the perceptron computed on a subset represents a true hypothesis on a part of the entire sample set. The different hypotheses are then taken together by a voting procedure to form the final hypothesis. This approach goes along the lines of a learning method which has been called Epicurean learning by Cleary et al. in [9] (with reference to [24]), motivated by Epicurus' paradigm that all hypotheses fitting the known data should be retained [12]. Furey et al. [11] used a similar method when they compared SVM classifications of gene-expression data to a perceptron-based learning algorithm (referring to the approach of taking linear combinations of decision rules obtained at each iteration step as described in [15]). We provide results from computational experiments showing that the classification rate becomes worse if the size of randomly chosen subsets becomes close to the entire sample set, although different hypotheses are calculated by the underlying stochastic local search.

2 Methods

Let $D \subseteq \mathbb{Q}^n$ be our input data table where each of the columns corresponds to expression measurements for a particular gene over the tissues investigated. Further let $c : \mathbb{Q}^n \rightarrow \{1, 2, \dots, m\}$ be a partial function that for D returns the tumor class associated with each row.

We would like to find a realization of a function $F : \mathbb{Q}^n \rightarrow 2^{\{1, 2, \dots, m\}}$ that

represents an extension of c that we can use to classify new, unseen expression measurement vectors.

We do this as follows. For each class $i \in \{1, 2, \dots, m\}$, we construct a classifier $F_i : \mathbb{Q}^n \rightarrow [0, 1]$. These F_i are then combined to form F as:

$$F(x) = \{j | F_j(x) = \max_i F_i(x)\}.$$

The number $|F(x)|$ gives us an indication of how uniquely we were able to classify x . We choose to discard the classification if $|F(x)| > 1$.

We now turn to the construction of the functions F_i . A *perceptron* p is a function $p : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \rightarrow \{0, 1\}$ such that

$$p(x, w, \vartheta) = p_{w, \vartheta}(x) = \tau_{\vartheta}(wx),$$

where τ is the unit step function at threshold ϑ defined as

$$\tau_{\vartheta}(y) = \begin{cases} 0 & \text{if } y < \vartheta, \\ 1 & \text{otherwise.} \end{cases}$$

Given k_i perceptrons, we define F_i to be

$$F_i(x) = \frac{1}{k_i} \sum_{j=1}^{k_i} p_{w, \vartheta}^{ij}(x).$$

where the w 's are restricted to be rational (for simpler notations, we assume $\vartheta = 0$ and always $x_n = 1$, i.e., w_n represents the threshold). There are two problems that need to be solved:

1. Finding the w_j^i 's, i.e., training the perceptrons;
2. Finding k_i for each class i .

The parameter k_i is chosen empirically. We will discuss how this is done in Section 5. In the remainder of this section we address the training of the perceptrons, and pragmatics regarding reduction of the input dimensionality.

2.1 Perceptron Training

Let $T = T^+ \cup T^-$ be a training set composed of positive and negative examples, respectively. We want to find the parameters w of the perceptron p that maximize the separation of the positive and negative examples in T .

Höfgen and Simon [16] have shown that finding a linear threshold function that minimizes the number of misclassified vectors is NP-hard in general. Hence we need to apply heuristics to the problem of training a perceptron.

Simulated annealing has been proven to be a versatile tool in combinatorial optimisation [1, 2, 3, 4, 18], and is our choice of optimization strategy. In [7], we have demonstrated on randomly generated disjunctive normal forms that the classification rate improves if the perceptron algorithm is combined with logarithmic simulated annealing.

Given a search space W in which we want to find an element that minimizes an objective function $o : W \rightarrow \mathbb{N}$, an initial “current location” $w_s \in W$, a function $s : W \rightarrow W$ that in a stochastic fashion proposes a next “current location” that we hope will lead to the wanted optimum, a function $a : \mathbb{N} \times W \times W \rightarrow \{0, 1\}$ that accepts or rejects the proposed next current location, and finally a stopping criterion. We can illustrate the strategy by the following simple pseudo-code skeleton:

```

 $w \leftarrow w_s; k \leftarrow 0$ 
while not stop( $k, w$ )
   $k \leftarrow k + 1; w_n \leftarrow s(w)$ 
  if  $a_k(w, w_n) = 1$ 
     $w \leftarrow w_n$ 

```

The idea of the algorithm is to, while initially allowing locally suboptimal steps, become more restrictive in allowing locally sub-optimal steps to be taken over time. The hope is to avoid premature convergence to local minima. In our case we define our objective function to be the number of misclassified elements in T . Let

$$M_T(w) = \{x \in T^+ | p_{w,0}(x) < 1\} \cup \{x \in T^- | p_{w,0}(x) > 0\},$$

then we can define our objective function as $o(w) = |M_T(w)|$. The set $M_T(w)$ can be viewed as a neighborhood of w containing all the possible next steps we can take from w . As a *first key feature* of our heuristic, we now construct a probability mass function u over $M_T(w)$ as

$$q(x) = |xw| / \sum_{y \in M_T(w)} |yw|.$$

Elements that are “further away” from being classified correctly by $p_{w,0}$ are assigned higher values by q . We now define

$$s(w) = w - \chi(x) \cdot \text{sample}(M_T(w), q) / \sqrt{w},$$

where sample stochastically selects one element from the set $M_T(w)$ with the probability given by q , and $\chi(x) = 1$ for $x \in T^-$, $\chi(x) = -1$ for $x \in T^+$. The acceptance function at step k (the k 'th time through the while-loop) is defined as

$$a(w_{k-1}, w_k) = \begin{cases} 1 & \text{if } \pi(w_{k-1}, w_k) > \rho, \\ 0 & \text{otherwise,} \end{cases}$$

where

$$\pi(w_{k-1}, w_k) = \begin{cases} 1 & \text{if } o(w_k) - o(w_{k-1}) \leq 0, \\ e^{-(o(w_k) - o(w_{k-1}))/t(k)} & \text{otherwise,} \end{cases}$$

and $\rho \in [0, 1]$ is uniformly randomly sampled at each step k . The function t , motivated by Hajek's Theorem[14] on convergence of inhomogenous Markov chains for big enough constants Γ , is defined as

$$t(k) = \Gamma / \ln(k + 2), \quad k \in \{0, 1, \dots\},$$

and represents the “annealing” temperature (*second key feature*). As t decreases, the probability of accepting a w that does not decrease the objective function decreases. We empirically chose $\Gamma = (|T^+| + |T^-|)/3$, essentially using the same method as in [5].

Finally our stopping criterion is given by a pre-determined number of iterations through the while-loop.

2.2 Perceptron Training Set Sampling

In order to generate (a large number of) different hypotheses as well as to achieve zero learning error in a short time, the following sampling scheme for training sets is applied (*third key feature*):

Let $C_i^+ = \{x \in D | c(x) = i\}$, be the positive examples for class i in D , and let $C_i^- = D - C_i^+$, be the negative examples of class i in D . Further, let for two parameters $\alpha, \beta \in (0, 1]$, $s_i^+ = \lfloor \alpha |C_i^+| \rfloor$, and let $s_i^- = \lfloor \beta |C_i^-| \rfloor$. For each j in $\{1, 2, \dots, k_i\}$ we randomly sample $T_{i,j}^+ \subseteq C_i^+$ and $T_{i,j}^- \subseteq C_i^-$ such that $|T_{i,j}^+| = s_i^+$, and $|T_{i,j}^-| = s_i^-$. The set $T_{i,j} = T_{i,j}^+ \cup T_{i,j}^-$ is then the training set used to train perceptron p_j in F_i .

The Boosting Method (cf. [23, 24]) tries to reduce the training error by assigning higher probabilities to “difficult” samples in a recursive learning procedure. In our approach, we observed that almost all subsets are learned with zero error even for relatively large fractions of the entire sample set (about 0.5 in [5] and 0.75 in the present study). Thus, according to the procedure described in the previous section, the method we are using belongs to the class of voting predictions. Recent research has shown that classification performance can be significantly improved by voting on multiple hypotheses [24]; for a more detailed discussion cf. [9]. We note that the particular examples are trained multiple times, e.g., for the ALL/AML data set, the average occurrence of an ALL sample in randomly chosen subsets is about 220 in trials with the best classification rate.

2.3 Dimensionality Reduction

In our case where the data D presents a massively overdetermined system, i.e., there are many more gene expression measurements than there are tissue samples, experiments have shown that reducing the dimensionality of the data is beneficial [17]. The scheme we applied is based on selecting the genes that incur the coefficients with the biggest absolute values in the perceptrons after having completed training on all dimensions in the data (*fourth key feature*). Let $g(w, q)$ be the set of q positions that produce the q biggest values $|w_l|$ in $w = (w_1, w_2, \dots, w_n)$ (ties are ordered arbitrarily). Let $G_i = \bigcap_{j=1}^p g(w_j^i, q)$ be the set of dimensions selected for class i , i.e., here we have $k_i = p$ for all $i = 1, \dots, m$. Each G_i is truncated to $\kappa := \min_{i \in \{1, \dots, m\}} |G_i|$ positions with the largest associated values (called priority genes). Training each F_i is then repeated with the data D projected onto the dimensions in G_i for $k_i = K$

perceptrons, $i = 1, \dots, m$. The importance of weight size in learning procedures has been emphasised by P. Bartlett in [8].

3 Data Sets

Improvements in cancer classification have been central to advances in cancer treatment [13]. Usually, cancer classification has been based primarily on the morphological appearance of the tumour which has serious limitations: Tumours with similar appearance can follow significantly different clinical courses and show different responses to therapy. In a few cases, such clinical heterogeneity has been explained by dividing morphologically similar tumours into subtypes. Key examples include the subdivision of small round blue cell tumours and acute leukemias. Both tumour classes are considered in the present section.

3.1 SRBCT Data

For the first series of computational experiments, the data used in this paper are provided by Khan et al. in [17]. Given are gene-expression data from cDNA microarrays containing 2308 genes for four types of small, round blue cell tumours (SRBCTs) of childhood, which include neuroblastoma (NB), rhabdomyosarcoma (RMS), Burkitt lymphoma (BL) and the Ewing family of tumours (EWS), i.e., here we have $m = 4$. The number of training samples is as follows: 23 for EWS, 8 for BL, 12 for NB, and 20 for RMS. The test set consists of 25 samples: 6 for EWS, 3 for BL, 6 for NB, 5 for RMS, and 5 "others". The split of the data into training and test sets was the same as in the paper by Khan et al., where it has been shown that a system of artificial neural networks can utilize gene expression measurements from microarrays and classify these tumours into four different categories. In [17], 3750 ANNs are calculated to obtain 96 genes for training the final ANN which is able to classify correctly the 20+5 test data.

3.2 AML/ALL Data

The data are taken from Golub et al. [13]. The training set consists of 7129 gene-expression data for 11 samples of acute myeloid leukemia (AML) and 27 samples of acute lymphoblastic leukemia (ALL), respectively (i.e., $m = 2$ in this case). For the test, 14 AML samples and 20 ALL samples are used (again, each of them is represented by 7129 gene-expression data). At present, there is no single test available to establish the distinction between AML and ALL. Current clinical practice involves an experienced specialist's interpretation of the tumour's morphology, histochemistry, immunophenotyping, and cytogenetic analysis, each performed in a separate, highly specialised laboratory [13]. To distinguish ALL from AML is critical for treatment. Golub et al. analysed various aspects of cancer classification in [13]. In particular, by using the model

of self-organising maps, Golub et al. obtained a strong prediction for 18 out of the 20 ALL test samples and 10 out of the 14 AML samples.

4 Results

The algorithm described in Section 2 has been implemented in C^{++} and we performed computational experiments for the data sets from Section 3 on SUN Ultra 5/333 workstation with 128 MB RAM. For both data sets, we present three types of results from computational experiments.

4.1 SRBCT Data

In Table 1, computations include all 2308 input variables (gene-expression data). The parameter settings are $\alpha = 0.75$ and $\beta = 0.25$ for balanced but still sufficiently large numbers of positive and negative examples. The entries in the table are the errors on classes in the order [EWS,BL,NB,RMS]. We recall that the number K of threshold functions (perceptrons) is the same for all four classes, i.e., $K = k_i$, $i = 1, \dots, 4$. The values are typical results from three to five runs for each parameter setting; the deviation is very small and therefore average values are not calculated.

K	11	33	99	297	891
Error Distr.	[1,1,5,0]	[1,1,5,0]	[0,1,5,0]	[0,1,5,0]	[0,1,5,0]
Total errors	35%	35%	30%	30%	30%

Table 1

Classification results when training performed on all 2308 genes.

In Table 2, the training procedure has been performed on priority genes only, i.e., on κ genes that are determined by the intersection of p sets (derived from p perceptrons) consisting of q most significant weights.

The parameters are $p = 5, 9, 11$ and $q = 250$ (q is large enough to have non-empty intersections), and the numbers of priority genes are $\kappa = 23, 10, 8$. The results are stable for values close to $p = 9$ and larger $K \geq 300$, i.e., the error is equal to zero if $p \in \{8, 9, 10\}$ and $K \gg 300$. While the top rating for each example is in general at least twice as large as the second largest one (for most cases much higher), the rating for the test example No. 20 is only marginally better than the second best rating. This example causes the misclassification on EWS for $p = 11$ and for the first three entries of $p = 9$. On this example, Kahn et al. report a vote of 0.40 [17], and our ratings are in the same range between 0.41 to 0.45.

p / K	11	33	99	297	891
5 ($\kappa = 23$)	[1,0,4,0]	[1,0,4,0]	[0,0,4,0]	[0,0,4,0]	[0,0,4,0]
Total errors	25%	25%	20%	20%	20%
9 ($\kappa = 10$)	[1,0,2,0]	[1,0,2,0]	[1,0,0,0]	[0,0,0,0]	[0,0,0,0]
Total errors	15%	15%	5%	0%	0%
11 ($\kappa = 8$)	[1,0,1,0]	[1,0,1,0]	[1,0,0,0]	[1,0,0,0]	[1,0,0,0]
Total errors	10%	10%	5%	5%	5%

Table 2

Classification results when training performed on κ priority genes only.

The rating of the five "other" examples is below 0.25. Thus, our method is sensitive to the right choice of p . We obtain zero classification error on 10 genes only, whereas in [17] 96 genes are used for classification (which are derived from 3750 ANNs; the ANNs are trained and evaluated on different selections out of the sample set).

The run-time ranges from 1 *min* (for $p = 5$ and $K = 11$) up to 113 *min* (for $p = 11$ and $K = 891$).

Since we are using $|w_l|$ for the selection of priority genes, it is interesting to know whether the largest absolute values of weights correspond to the most significant average values of gene-expression data. Table 3 provides the rank of the average value for priority genes calculated for $K = 297$ and the EWS cancer type. In this run, we have $p = 9$ and $\kappa = 10$ (cf. Table 2). We recall that $q = 250$; for the largest average values of gene-expression data this means the range of ranks 2059, ..., 2308.

Gene	246	469	509	545	1708	1781	1834	1841	1961	2223
Rank	2280	2290	1580	2284	2186	2306	2166	1770	1544	2049

Table 3

Ranking of Priority Genes for SRBCT data.

In Table 3, 40% of the genes do not belong to this range; thus, there seems to be no direct correspondence between the rank of the average value of gene-expression data and the absolute value of weights in classifying threshold functions from the first layer of the depth-three circuit, although all ten ranks belong to the upper half of rank values.

For $p = 9$ ($\kappa = 10$) and $K = 297$ we investigated the impact of the random choice of subsets from the entire sample set. Since the sample sets are relatively small, the parameter settings $\alpha = 0.75$ and $\beta = 0.25$ were chosen in our basic experiments displayed in Table 2. We performed experiments for $\alpha = 0.85, \dots, 1.0$ and $\beta = 0.35, \dots, 0.5$. The results are shown in Table 4.

Our algorithm represents a stochastic search procedure, and therefore it produces (in general) different hypotheses even for the case $\alpha = \beta = 1.0$. To have a

balanced number of positive and negative examples, we chose $\alpha = 1.0$, $\beta = 0.5$ as maximum values. Table 4 clearly demonstrates the effect of Epicurean learning: The classification results become worse if the number of potential training subsets becomes smaller.

$p / [\alpha, \beta]$	$[0.80, 0.30]$	$[0.85, 0.35]$	$[0.90, 0.40]$	$[0.90, 0.40]$	$[1.0, 0.5]$
9 ($\kappa = 10$)	$[0, 0, 0, 0]$	$[0, 0, 2, 0]$	$[0, 0, 3, 0]$	$[0, 0, 3, 0]$	$[0, 0, 3, 0]$
Total errors	0%	15%	15%	15%	15%

Table 4

Impact of Epicurean Learning (SRBCT data).

4.2 AML/ALL Data

In this case, we have a binary classification problem and therefore $\alpha = \beta = 0.75$ has been chosen. In Table 5, the training has been performed on all 7129 input variables (gene-expression data). The entries in the table are the errors in the order [ALL,AML]. The values are typical results from several runs for each parameter setting.

The classification improves on the results published in [13]. The run-time is between 1 *min* and 44 *min*. The problem is whether the same or even better results are possible on significantly smaller numbers of gene data inputs.

K	33	99	297	891	2673
Error Distr.	$[0, 3]$	$[1, 2]$	$[1, 2]$	$[2, 2]$	$[2, 1]$
Total errors	8.8%	8.8%	8.8%	11.8%	8.8%

Table 5

Classification results when training performed on all 7129 genes.

In Table 6, the training procedure has been performed on priority genes only, according to the procedure described in Section 2.3. Here, the parameters are $p = 9, 13, 17$ and the same $q = 250$ as for SRBCT data. The corresponding numbers of priority genes are $\kappa = 69, 48, 42$. The run-time ranges from 13 *min* (for $p = 9$ and $K = 33$) up to 578 *min* (for $p = 17$ and $K = 2673$).

For $p = 13, 17$ and $K \geq 297$, we have two classification errors, whereas the best result reported in [13] are six errors. Moreover, our classification is obtained on 42 genes only ($p = 17$).

If $K \ll 300$ for $p = 17$, the number of priority genes seems to be too small to provide a better classification, whereas $\kappa = 69$ ($p = 9$) is too large in order to achieve the same result as in Table 5 where probably the impact of “noisy data” cancels out on a larger number of input data.

p / K	33	99	297	891	2673
9 ($\kappa = 69$)	[2,2]	[2,2]	[2,2]	[2,2]	[2,2]
Total errors	11.8%	11.8%	11.8%	11.8%	11.8%
13 ($\kappa = 48$)	[1,1]	[1,2]	[1,1]	[1,1]	[1,1]
Total errors	5.9%	8.8%	5.9%	5.9%	5.9%
17 ($\kappa = 42$)	[2,1]	[2,1]	[1,1]	[1,1]	[1,1]
Total errors	8.8%	8.8%	5.9%	5.9%	5.9%

Table 6

Classification results when training performed on κ priority genes only.

The classifications shown in Table 6, the ratings are strict (i.e., above the threshold 0.5) on almost all samples, except for the two samples ALL-43 and AML-64. On these two samples, the classification is usually in favour of the correct prediction, but the margin between the ratings is very small, e.g., [0.48,0.5] for AML-64 and [0.35,0.32] for ALL-43.

In Table 7, we provide the rank of the average value for priority genes calculated for $K = 297$ and $p = 13$ ($\kappa = 48$). Since we have $q = 250$ again, the largest average values of gene-expression data correspond to the rank numbers 6880, ..., 7129. More than 58% of the genes from Table 7 do not belong to this range.

Gene	19	45	532	760	1092	1133	1239	1249	1376	1400
Rank	7104	7056	7004	2429	5388	5396	6907	6260	6903	5924
Gene	1421	1694	1704	1882	1928	2043	2094	2121	2186	2288
Rank	6969	6943	6980	4127	6905	4272	6967	6642	6846	3150
Gene	2402	2501	2597	3056	3258	4136	4142	4196	4377	4535
Rank	5598	6625	6971	6621	6551	6939	6763	6596	5350	6507
Gene	4654	4847	5191	5507	5552	5642	5711	5716	5772	6180
Rank	7007	4896	6748	7111	6976	6972	7068	7048	6908	6110
Gene	6200	6201	6209	6345	6760	6797	6803	6806		
Rank	4648	6053	7052	6593	6977	6552	6566	6517		

Table 7

Ranking of Priority Genes for ALL/AML data.

As for SRBCT data, we analysed the impact of an increasing size of randomly chosen subsets of the entire sample set. For $p = 13$ ($\kappa = 48$) and $K = 297$ we performed computational experiments for $\alpha = \beta = 0.85, \dots, 1.0$.

As can be seen from Table 8, the classification becomes worse as the number of potential training subsets decreases, if compared to the results from Table 6.

$p / [\alpha, \beta]$	[0.80,0.80]	[0.85,0.85]	[0.90,0.90]	[0.95,0.95]	[1.0,1.0]
13 ($\kappa = 48$)	[2,1]	[2,1]	[3,1]	[3,1]	[3,1]
Total errors	8.8%	8.8%	11.8%	11.8%	11.8%

Table 8

Impact of Epicurean Learning (ALL/AML data).

5 Discussion

Initial reports on the analysis of gene expression data derived from microarrays concentrated on unsupervised learning, in which the main objective was to determine which genes tend to have correlated patterns of expression. In early experiments, the number of actual arrays was usually small, especially when compared to the number of measurements per case, which is generally in the order of thousands. As microarray technology evolves and becomes less expensive, the number of arrays tends to grow, allowing for the construction of supervised learning models. Supervised learning models are increasingly being used, and classification of cases into diagnostic or prognostic categories has been attempted. Besides classification performance, an important goal of these models is to determine whether a few genes can be considered good markers for disease. It is important to investigate whether simple learning models can be successfully used for this purpose. In this paper, we showed an algorithm based on simulated annealing that is used to train a system of perceptrons, each obtained by resampling of the training set. The model was able to successfully classify previously unseen cases of SBRCTs using a small number of genes. Furthermore, we provided experimental evidence that Epicurean-style learning might be essential to obtain satisfactory classification results.

References

- [1] E.H.L. Aarts. *Local Search in Combinatorial Optimization*. Wiley&Sons, 1998.
- [2] A. Albrecht, S.K. Cheung, K.S. Leung, and C.K. Wong. Stochastic Simulations of Two-Dimensional Composite Packings. *Journal of Computational Physics*, 136(2):559–579, 1997.
- [3] A. Albrecht, S.K. Cheung, K.S. Leung, and C.K. Wong. Computing Elastic Moduli of Two-Dimensional Random Networks of Rigid and Nonrigid Bonds by Simulated Annealing. *Mathematics and Computers in Simulation*, 44(2):187–215, 1997.
- [4] A. Albrecht, S.K. Cheung, K.S. Leung, and C.K. Wong. On the Convergence of Inhomogeneous Markov Chains Approximating Equilibrium Placements of Flexible Objects. *Computational Optimization and Applications*, 19(2):179–208, 2001.

- [5] A. Albrecht, E. Hein, K. Steinöfel, M. Taupitz, and C.K. Wong. Bounded-Depth Threshold Circuits for Computer-Assisted CT Image Classification. *Artificial Intelligence in Medicine*, 24(2):177-190, 2002.
- [6] A. Albrecht, K. Steinhöfel, M. Taupitz, and C.K. Wong. Logarithmic Simulated Annealing for Computer-Assisted X-ray Diagnosis. *Artificial Intelligence in Medicine*, 22(3): 249–260, 2001.
- [7] A. Albrecht and C.K. Wong. Combining the Perceptron Algorithm with Logarithmic Simulated Annealing. *Neural Processing Letters*, 14(1):75-83, 2001.
- [8] P. Bartlett. The Sample Complexity of Pattern Classification with Neural Networks: The Size of Weights is more Important than the Size of the Network. *IEEE Transactions on Information Theory*, 44(2):525-536, 1998.
- [9] J.G. Cleary, L.E. Trigg, G. Holmes, and M.A. Hall. Experiences with a Weighted Decision Tree Learner. In: M Bramer, A Preece, and F Coenen, eds., *Research and Development in Intelligent Systems XVII*, pp. 35–47, BCS Series, Springer-Verlag, 2000.
- [10] M.B. Eisen, P.T. Spellman, P.O. Brown, D. Botstein. Cluster Analysis and Display of Genome-wide Expression Patterns. *Proc. Natl. Acad. Sci. USA*, 95(25):14863-8, 1998.
- [11] T.S. Furey, N. Cristianini, N. Duffy, D.W. Bednarski, M. Schummer, and D. Haussler. Support Vector Machine Classification and Validation of Cancer Tissue Samples Using Microarray Expression Data. *Bioinformatics*, 16:906-914, 2000.
- [12] C.-F. Geyer. *Epikur*. Junius-Verlag, Hamburg, 2000.
- [13] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander. Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science*, 286:531–537, 1999.
- [14] B. Hajek. Cooling Schedules for Optimal Annealing. *Mathem. of Operations Research*, 13:311 – 329, 1988.
- [15] D. Helmbold and M.K. Warmuth. On Weak Learning. *J. of Computer and System Sciences*, 50:551-573, 1995.
- [16] K.-U. Höffgen, H.-U. Simon, and K.S. van Horn. Robust Trainability of Single Neurons. *J. of Computer System Sciences*, 50:114–125, 1995.
- [17] J. Khan, J.S. Wei, M. Ringner, L.H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C.R. Antonescu, C. Peterson, and P.S. Meltzer. Classification and Diagnostic Prediction of Cancers Using Gene Expression Profiling and Artificial Neural Networks. *Nature [Medicine]*, 7(6):673-679, 2001.
- [18] S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671 – 680, 1983.
- [19] M.L. Minsky and S.A. Papert. *Perceptrons*. MIT Press, Cambridge, Mass., 1969.
- [20] N.J. Maughan, F.A. Lewis, and V. Smith. An Introduction to Arrays. *J. of Pathology*, 195:3-6, 2001.

- [21] J. Quackenbush. Computational Analysis of Microarray Data. *Nature Reviews [Genetics]*, 2(6):418-427, 2001.
- [22] F. Rosenblatt. *Principles of Neurodynamics*. Spartan Books, New York, 1962.
- [23] R.E. Schapire. The Strength of Weak Learnability. *Machine Learning*, 5(2):197-227, 1990.
- [24] R.E. Schapire, Y. Freund, P. Bartlett, and W.S. Lee. Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods. *The Annals of Statistics*, 26(5):1651-1686, 1998.