2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Supervised Classification and Gene Selection Using Simulated Annealing

Maurizio Filippone, Francesco Masulli, Stefano Rovetta
DISI - Dipartimento di Informatica e Scienze dell'Informazione, Università di Genova and CNISM
Via Dodecaneso 35, 16146 Genova (Italy) – {*filippone, masulli, rovetta*}*@disi.unige.it*

***Abstract*— Genomic data are often characterized by small cardinality and high dimensionality. For those data, a feature selection procedure could highlight the relevant genes and improve the classification results. In this paper we propose a wrapper approach to gene selection in classification of gene expression data using Simulated Annealing and SVM. The proposed approach can do global combinatorial searches through the space of possible input subsets, can handle cases with numerical, categorical or mixed inputs, and is able to find (sub-)optimal subsets of input variables giving very low classification errors. The method has been tested on the publicly available data sets Leukemia by Golub et al. and Colon by Alon at al. The experimental results highlight the capacity of the method to select minimal sets of relevant genes.**

I. Introduction

Genomic data are often characterized by small cardinality and high dimensionality and can include some features that are not relevant for the discrimination among classes. This is the case e.g. with gene expression data obtained from DNA microarrays where each dimension or feature corresponds to a gene expression data. Usually, some (or most) genes are not relevant for the discrimination between classes. This is inherent in the experiment design: a lot of candidate genes are probed in a microarray experiment, and those related to the phenomenon under study are to be identified.

For those data, a gene selection procedure could highlight the relevant genes and improve the classification results at the same time.

In this paper we propose a wrapper approach to gene selection in classification of gene expression data. The combinatorial search is performed using the Simulated Annealing (SA) method [14] which is a global search method technique derived from Statistical Mechanics and based on the Metropolis algorithm [18], while the learning algorithm is the SVM which is one of the most popular classifiers [6].

In the next section we briefly review the input variable selection problem and approaches, then the Simulated Annealing technique is presented and we describe how we applied it to input selection problem (Sect. III) with a mechanism for the measurement of the input relevance (Sect. IV). In Sect. V and VI the experimental validation of the method is shown and Sect. VII concerns the conclusions.

II. Gene selection as a variable selection problem

Gene selection is a specific instance of a more general problem, which is called input or variable selection [12]. Note that our main goal is selection of relevant genes rather than performance optimization or dimensionality reduction. Therefore *features* (derivative variables obtained by transformation of raw input variables) are not of interest. Input selection algorithms can be broadly divided into two categories [4], [15]: filters and wrappers. Filters evaluate the relevance of each input (subset) using the data set alone, while wrappers invoke a learning algorithm to evaluate the quality of each input (subset). Both approaches, filters and wrappers, usually involve combinatorial searches (often only local) through the space of possible input subsets. Wrappers are usually more computationally demanding, but they can be superior in accuracy when compared with filters.

The strategy for variable selection, and the underlying assumptions about the input variables themselves, is also a design choice. Variables can be selected as a subset with aggregate discriminative power [22], [20], or ranked for their individual relevance [24], [17]. In the latter case variables are assumed to be weakly correlated, so that their individual importance can be unambiguously assessed. In the former case, it is assumed that all possible interaction patterns can occur, and this forces a much more complex search space. However, ranks can also be used as an indication to evaluate a subset selection process, in an in-between approach.

The definition of relevance itself can be subject to variations [15], and the goal of the procedure can also be different, with some approaches aiming at comprehensive set (find all significant variables [13], [16], [20]) or at explanatory sets (this is generally the case with all gene selection tasks, where one wants to identify the most important genes only, as e.g. in [10]). Again, an in-between approach is possible [24] when an explicit cost function includes both performance and complexity (number of variables) terms. In this case, a continuum of possible balances is provided by the relative weighting of these terms.

To summarize, the gene selection problem is stated as the problem of selecting small subsets of input variables achieving high discriminating power but with good explanatory properties. Interactions among genes may and do occur, but as a working hypothesis it is possible to rank genes according to their individual relevance. These hypotheses form the basis of the method we are presenting here, which is based on optimizing the combination of performance and complexity costs.

0-7803-9490-9/06/$20.00/©2006 IEEE

TABLE I

SIMULATED ANNEALING INPUT SELECTION (SAIS) ALGORITHM.

1) Initialization of parameters;
2) Initialize $\mathbf{g}$ at random (binary mask);
3) Perform classification and evaluate the generalized system energy E;
4) **do**
5) Initialize $f = 0$ (number of iterations), h=0 (number of success);
   a) **do**
   b) Increment number of iterations f;
   c) Perturb mask $\mathbf{g}$;
   d) Perform classification and evaluate the generalized system energy E;
   e) Generate a random number rnd in the interval [0,1];
   f) **if** $rnd < P(\Delta E)$ **then**
      i) Accept the new $\mathbf{g}$ mask;
      ii) Increment number of success h;
   g) **endif**
   h) **loop until** $h \leq h_{min}$ **and** $f \leq f_{max}$;
6) update $T = \alpha T$;
7) **loop until** $h > 0$;
8) end.

III. SA FOR GENE SELECTION

The method for input selection we propose makes use of Simulated Annealing (SA) technique [14] that is a global search method technique derived by Statistical Mechanics. SA is based on the Metropolis algorithm [18] that has been proposed to simulate the behavior and small fluctuations of a system of atoms starting from an initial configuration, by the generation of a sequence of iterations. In the Metropolis algorithm each iteration is composed by a random perturbation of the actual configuration and the computation of the corresponding energy variation (ΔE). If $\Delta E < 0$ the transition is unconditionally accepted, otherwise the transition is accepted with probability given by the Boltzmann distribution:

$$P(\Delta E) = exp\left(\frac{-\Delta E}{KT}\right) \tag{1}$$

where K is the Boltzmann constant and T the temperature.

In SA this approach is generalized to the solution of general optimization problems [14] by using an *ad hoc* selected cost function (*generalized energy*), instead of the physical energy. SA works as a probabilistic hill-climbing procedure searching for the global optimum of the cost function. The temperature T takes the role of a control parameter of the search area (while K is usually set to 1), and is gradually lowered until no further improvements of the cost function are noticed. SA can work in very high-dimensional searches, given enough computational resources.

SA has been already applied to classification of gene expression data from DNA microarray [1] with the aim of training perceptrons.

In this paper we apply SA to the variable selection problem. The approach we adopted is to constrain the search space to subsets of variables, and to evaluate a compound cost function combining performance and complexity scores, as previously indicated. The method is described in the following, whereas in Tab. I a step-by-step outline of the proposed Simulated Annealing Input Selection (SAIS) algorithm is presented.

Let $\mathbf{g} = (g_1, g_2, \ldots, g_d)$ be a binary mask representing the system state (configuration), where each bit g_i (with $i = 1, \ldots, d$) corresponds to the selection ($g_i = 1$) / deselection ($g_i = 0$) of a input (if we want to select a set of s inputs, at each time only s bits will be set to 1). The initialization of the vector mask $\mathbf{g}$ (Step 2) is done by generating s_0 integer numbers with uniform distribution in the interval $[1, d]$ and setting the corresponding bits to 1 of $\mathbf{g}$ and the remaining ones to 0. A perturbation or move (Step 5c) is done in the following way:

1) chose w and v following uniform distributions, respectively in the intervals $[w_{\min}, w_{\max}]$ and $[v_{\min}, v_{\max}]$;
2) a number of w of genes set to 1 are set to 0;
3) a number of v of genes set to 0 are set to 1;

The values $w_{\min}, w_{\max}, v_{\min}, v_{\max}$ regulate the variability of each perturbation.

The classification task (Steps 3 and 5d) is performed in the sub-space of selected inputs defined by the vector mask $\mathbf{g}$. After each run of the classification algorithm we can obtain an evaluation of the generalized energy E as a linear combination of the *Classification Error* ε (obtained with a cross-validation technique, such as *leave one out* or *k-fold validation*) and of the number of selected genes s:

$$E = \varepsilon + \lambda\, s \tag{2}$$

The introduction of the number of selected genes s in the computation of E penalizes situations in which the number of selected genes is too high. This choice of E leads to the minimization of the number of genes able to achieve a good classification error. The compromise between these two terms is controlled by λ (*regularization coefficient*).

The initial value of temperature T is obtained as the average value of ΔE computed over an assigned number p of random perturbations of the mask $\mathbf{g}$.

SAIS is a computational intensive algorithm, but it is able to work with both numeric and categorical variables. It is worth noting that each time we run the SAIS algorithm we can find a sub-optimal subset of s inputs from the original d. In principle, each independent run of SAIS can lead to a quite different subset of inputs.

IV. RANKING INPUT RELEVANCE

SA is an algorithm implementing a stochastic time-varying dynamical system where the state vector evolves in the direction of the minima of the generalized energy function. In our case during the evolution of the SAIS algorithm the bits set in the state vector $\mathbf{g}$ will be related to the more relevant inputs (genes) with increasing probability.

The inputs which are more relevant for classification should appear soon in the set of bits set to 1 and will be as more frequent as the temperature decreases. In order to estimate the relevance of inputs, we can include in the SAIS an aging algorithm. To this end, we can define a vector $\mathbf{r} = (r_1, r_2, \ldots, r_d)$. At Step 2 of the SAIS algorithm, we set

TABLE II
VALUES OF PARAMETERS

Meaning	Symbol	Experiment 1	Experiment 2
Number of random perturbations of $\mathbf{g}$ used to estimate the initial value of T	p	10000	10000
Number of inputs to be initially selected	s_0	20	20
Cooling parameter	α	0.9	0.9
Maximum number of iteration at each T	f_{max}	10000	2000
Minimum number of success for each T	h_{min}	1000	200
Regularization coefficient	λ	10^{-2}	$2 \cdot 10^{-3}$
Minimum number of bits to be switched	$w_{\min}$, $v_{\min}$	1	1
Maximum number of bits to be switched	$w_{\max}$, $v_{\max}$	s, 10	s, 10
Aging constant	γ	0.98	0.98

$r_i = 0 \ \forall i$. Every time a perturbation is accepted (Step 5.f), according to the Boltzmann distribution, we update $\mathbf{r}$ using this formula:

$$\mathbf{r} = \gamma \mathbf{r} + \mathbf{g} \tag{3}$$

where γ is the aging constant chosen in the interval [0,1].

At the end of the SAIS the vector $\mathbf{r}$ tells us how long each input has belonged to it in the last few successful moves of the algorithm. We give to vector $\mathbf{r}$ an interpretation as vector of input relevances.

It is worth noting that measures based on similar aging algorithms are also used in other dynamical systems such as operating systems of computers [21], sensor networks [9], and chaotic systems [3].

V. EXPERIMENTAL VALIDATION

SAIS has been implemented in R-language [19] under Linux operating system. We used as the classification algorithm the *Support Vector Machine* (SVM) [23] in the implementation of Chung et al. [5] that is one of most popular classifier. We chose to work with linear kernels and with a cost parameter $C = 1$ in the SVM functional in order to avoid model selection on more parameters and to make a biological interpretation of results possible.

The method was tested on two experiments on publicly available data sets that will be described in the next subsections.

On a Pentium IV 1900 MHz personal computer a complete running of SAIS on those data bases takes several hours (involving the run of hundreds of thousands of SVM).

A. First experiment

The first one is Leukemia data by Golub et al. [10] [1]. The Leukemia problem consists in characterizing two forms of acute leukemia, Acute Lymphoblastic Leukemia (ALL) and Acute Mieloid Leukemia (AML). The original work proposed both a supervised classification task ("class prediction") and an unsupervised characterization task ("class discovery").

[1] http://www.broad.mit.edu/cancer/software/genepattern/datasets/

The data set contains 38 samples for which the expression level of 7129 genes has been measured with the DNA microarray technique (the interesting human genes are 6817, and the other are controls required by the technique). These expression levels have been scaled by a factor of 100. Of these samples, 27 are cases of ALL and 11 are cases of AML. Moreover, it is known that the ALL class is in reality composed of two different diseases, since they are originated from different cell lineages (either T-lineage or B-lineage). In the data set, ALL cases are the first 27 objects and AML cases are the last 11. Therefore, in the presented results, the object identifier can also indicate the class (ALL if id ≤ 27, AML if larger). Using those data (with dimensionality $d = 7129$), Golub et al. [10] selected a set of 50 most relevant genes.

We applied the SAIS algorithm using a leave one out procedure for the computation of the classification error ε. The choice of the leave one out resampling technique is due to the scarcity of data in the sample.

In Fig. 1(a) and Fig. 1(b) the behavior of the classification error ε and the number of selected genes are plotted versus the iteration number of the algorithm in a run of SAIS. Each iteration corresponds to a different value of temperature T (i.e. Step 5 and Step 6 in Tab. I). This shows the ability of SAIS to minimize both the Classification Error ε and the number of relevant variables.

We did 10 independent runs of SAIS using the assumptions in Tab. II. For each run we obtained a set of two genes (even if we didn't require explicitly this number) with perfect discriminant ability ($\varepsilon = 0$). Each pair of selected genes (Tab. III) contains at least one gene found by Golub et al. [10]. Note that in Tab. III the classification error ε and the number of misclassified patterns m are also shown.

The genes ranked, using the definition of relevance of Sect. IV, in the first three positions are X95735_at, M23197_at and M55150_at. These three genes are also contained in the set selected by Golub et al. [10] and the most relevant gene X95735_at has been also highlighted by Guyon et al. [11].

TABLE III

GENES FOUND IN EACH RUN OF THE SAIS ALGORITHM ON THE LEUKEMIA DATA SET.

	Run 1	**Run 2**	**Run 3**	**Run 4**	**Run 5**
ε; m	0; 0	0; 0	0; 0	0; 0	0; 0
Gene 1	M55150_at	M58603_at	M23197_at	X95735_at	U29607_at
Gene 2	HG3523-HT4899_s_at	U50136_rna1_at	X85116_rna1_s_at	M10321_s_at	Y12670_at
	Run 6	**Run 7**	**Run 8**	**Run 9**	**Run 10**
ε; m	0; 0	0; 0	0; 0	0; 0	0; 0
Gene 1	M63138_at	D14659_at	M23197_at	HG3454-HT3647_at	M55150_at
Gene 2	U29091_at	X95735_at	M24470_at	X95735_at	HG3523-HT4899_s_at

TABLE IV

GENES FOUND IN EACH RUN OF THE SAIS ALGORITHM ON THE COLON DATA SET.

	Run 1	**Run 2**	**Run 3**	**Run 4**	**Run 5**
ε; m	0.048; 3	0.048; 3	0.048; 3	0.081; 5	0.048; 3
Gene 1	T95063	R38636	H20505	D12686	L22214
Gene 2	H07899	H67764	H05966	D21261	T69748
Gene 3	R49565	M95627	Z50753	H02630	Z50753
Gene 4	R01755	R36977	M94250	X12369	L06175
Gene 5	X79683	T40637	R54818	L10413	R74208
Gene 6	T74896	U05875	U31525	U10117	–
Gene 7	M22488	R90908	X66365	T56690	–
Gene 8	H81802	–	T51493	M38690	–
Gene 9	M24069	–	M83664	L14076	–
Gene 10	J02854	–	U33849	T47601	–
Gene 11	T67433	–	X56597	–	–
Gene 12	U09646	–	–	–	–
Gene 13	R49719	–	–	–	–
Gene 14	H13292	–	–	–	–
Gene 15	X77548	–	–	–	–
Gene 16	K03474	–	–	–	–
	Run 6	**Run 7**	**Run 8**	**Run 9**	**Run 10**
ε; m	0.032; 2	0.016; 1	0.065; 4	0.081; 5	0.016; 1
Gene 1	T54341	T74257	U37012	T74257	T74257
Gene 2	R87126	D26129	H64489	M76378	R37276
Gene 3	M95678	T81492	J03824	X13810	R07007
Gene 4	R27813	L09159	H65355	R49565	U37012
Gene 5	M11220	M23254	–	X05276	M23419
Gene 6	U30498	X78817	–	R55778	T78489
Gene 7	H05814	L06175	–	L37112	X66503
Gene 8	T86444	U27699	–	M23115	T89164
Gene 9	R88749	H08393	–	U33849	H09137
Gene 10	H49515	H06061	–	–	R36977
Gene 11	M63239	M84490	–	–	T89175
Gene 12	–	–	–	–	U07802
Gene 13	–	–	–	–	H49870
Gene 14	–	–	–	–	M13450
Gene 15	–	–	–	–	T85165
Gene 16	–	–	–	–	H56077
Gene 17	–	–	–	–	R54097
Gene 18	–	–	–	–	D14663
Gene 19	–	–	–	–	M28219
Gene 20	–	–	–	–	X73424
Gene 21	–	–	–	–	L06111

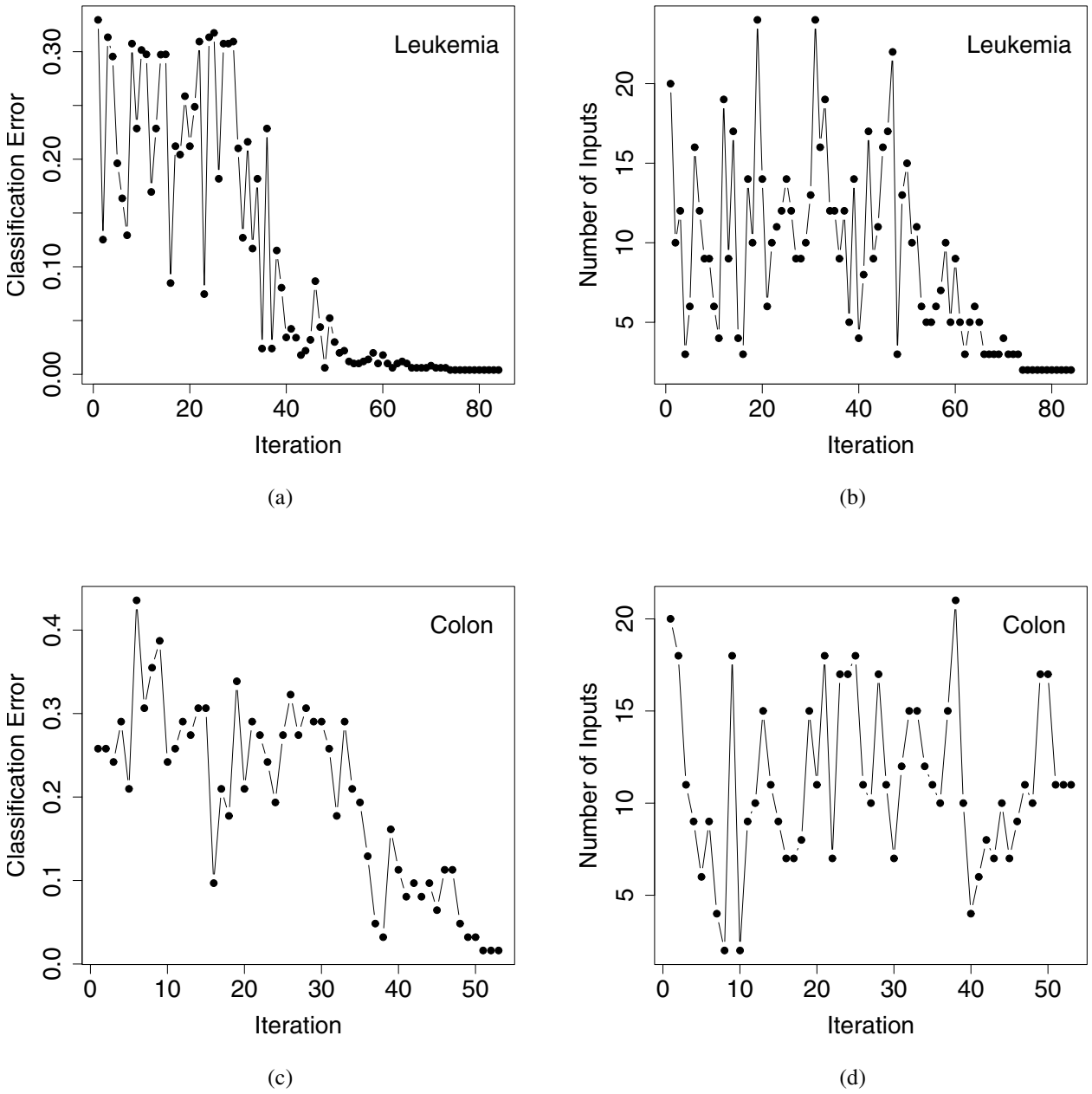

Fig. 1. Classification Error and Number of selected input variables vs the iteration number for Run 4 of Experiment 1 on Leukemia data set ((a) and (b)), and for Run 7 of Experiment 2 of Colon data set ((b) and (c)).

B. Second experiment

The second data set on which we performed input selection is the Colon data set by Alon et al. [2]. This is an oligonucleotide microarray analysis of gene expression in 40 tumor and 22 normal colon tissue samples, used to characterize the role and behavior of more than 6500 human genes in colon adenocarcinoma. The normal samples were obtained from a subset of the tumor samples, so that they are well paired to the corresponding positive samples. The actual data used in the experiments [2], contain only the 2000 most clearly expressed in the experiments, those with the highest minimal intensity across the 62 tissue samples.

In all the experiments on this data set, the classification error ε has been evaluated using k-fold validation with $k = 6$, as in this case we have a bigger (but still a small) data base than in previous experiment.

In Fig. 1(c) and Fig. 1(d) the behavior of the classification

[2]http://microarray.princeton.edu/oncology/affydata/index.html

error ε and the number of selected genes are plotted versus the iteration number of the algorithm.

We did 10 independent runs of SAIS using the assumptions in Tab. II. For each run we obtained a different set of 4-21 genes with good discriminant ability.

Run 7 selected 11 genes and made only one misclassification on the data base. Moreover, the gene H08393, obtained in this run, has been also highlighted by Guyon et al. [11].

VI. Conclusions

In this paper we have proposed a wrapper method for selecting inputs based on simulated annealing technique [14] and SVM [6]. The proposed approach performs a global combinatorial search in the space of input variables, allowing to select a minimal set of relevant inputs (genes), ranking at the same time their relevance.

On the 7129-dimensional Leukemia data set by Golub et al. [10] the proposed input selection method is able to find for each run a subset of two genes, that is sufficient to achieve null Classification Error with the leave one out procedure.

Different runs lead to obtain different pairs of genes and the more frequent ones are also in the list found by Golub.

On the 2000-dimensional Colon data set by Alon et al. [2] the proposed input selection method finds solutions with a low number of selected genes and a very good discriminant capability.

Note that each run of the algorithm yields a (possible new) minimal set of genes with the minimal classification error. The number of occurrence in the solution on the different runs or the medium value of relevance (evaluated as in Sect. IV) obtained in the different runs let us to rank the input variables.

Even if Simulated Annealing is computationally intensive SAIF algorithm shows a computational cost comparable with other wrapper methods. Let's consider, e.g., the method by Guyon et al. [11] which performs feature selection by recursive feature elimination. At each iteration a new linear SVM is trained and the input variable with the smallest weight is eliminated. In order to rank the entire set of thousand of features it must run an SVM for each dimension of the input space starting from thousands to one. In SAIF, instead, we have much more SVMs to train but each one performs classification in a space of small dimension (some tenths of input variables).

Moreover, it is worth noting that the proposed algorithm can work either with numerical or categorical variables depending on the associated learning machine and on the application. For example in [8] we run SAIS method associated to Fuzzy c-mean on the same data base obtaining minimal sets of inputs able to perform unsupervised clustering with null representation error.

Acknowledgment

Work funded by the Italian Ministry of Education, University and Research (2004 "Research Projects of Major National Interest", code 2004062740).

References

[1] Andreas Alexander Albrecht, Staal A. Vinterbo, and Lucila Ohno-Machado. An epicurean learning approach to gene-expression data classification. *Artificial Intelligence in Medicine*, 28(1):75–87, 2003.

[2] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, Proc. Natl. Acad. Sci. USA. vol.96 no.12 pp.6745–6750 (1999).

[3] E. Barkai, Aging in Subdiffusion Generated by a Deterministic Dynamical System, Phys. Rev. Lett. vol. 90, 104101 (2003)

[4] A. Blum and P. Langley, Selection of Relevant Features and Examples in Machine Learning, Artificial Intelligence, vol. 97, nos. 1–2, pp. 245–271, 1997.

[5] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/˜cjlin/libsvm.

[6] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273 – 297, 1995.

[7] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.

[8] Maurizio Filippone, Francesco Masulli, and Stefano Rovetta. Unsupervised gene selection and clustering using simulated annealing. In Isabelle Bloch, Alfredo Petrosino, and Andrea Tettamanzi, editors, *WILF*, volume 3849 of *Lecture Notes in Computer Science*, pages 229–235. Springer, 2005.

[9] D. Ganesan, B. Greenstein, D Perelyubskiy, D. Estrin and John Heidemann, An Evaluation of Multi-resolution Storage for Sensor Networks, Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003), 2003.

[10] Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C., Lander, E.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science* vol. 286, pp. 531–537, 1999.

[11] Guyon I., Weston J., Barnhill S., and Vapnik V. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[12] Isabelle Guyon, Andre Elisseeff. *An introduction to variable and feature selection*. *Journal of Machine Learning Research*, 3 1157–1182, 2003.

[13] K. Kira and L. Rendell, The Feature Selection Problem: Traditional Methods and a New Algorithm, Proc. 10th Nat l Conf. Artificial Intelligence (AAAI-92), pp. 129–134, 1992.

[14] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, vol. 220, pp.661–680, 1983.

[15] R. Kohavi and G. John, Wrappers for Feature Subset Selection, *Artificial Intelligence*, vol. 97, nos. 1–2, pp. 273–324, 1997.

[16] I. Kononenko, Estimating Attributes: Analysis and Extensions of RELIEF, Proc. Seventh European Conf. Machine Learning, pp. 171–182, 1994.

[17] F. Masulli and S. Rovetta, Random Voronoi ensembles for gene selection, *Neurocomputing*, vol. 55, no. 3-4, pp. 721-726, 2003.

[18] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equation of state calculations for fast computing machines. *Journal of Chemical Physics*, vol. 21, pp. 1087–1092, 1953.

[19] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. ISBN 3-900051-07-0. http://www.R-project.org

[20] C. Moneta, G.C. Parodi, S. Rovetta, and R. Zunino, Automated diagnosis and disease characterization using neural network analysis, in *Proceedings of the 1992 IEEE International Conference on Systems, Man and Cybernetics - Chicago, IL, USA*, pp. 123-128, 1992.

[21] A. Tanenbaum, Modern Operating Systems (2nd Edition), Prentice Hall, 2001.

[22] N. Slonim and N. Tishby, Agglomerative information bottleneck. In *Advances in Neural Information Processing Systems*, pages 617-623, 2000.

[23] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[24] J.Weston, A. Elisseff, B. Schoelkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, vol. 3, 1439-1461, 2003.