ISTITUTO DI ANALISI DEI SISTEMI ED INFORMATICA
"Antonio Ruberti"
CONSIGLIO NAZIONALE DELLE RICERCHE

E. Weitschek, G. Felici, P. Bertolazzi

MICROARRAY LOGIC ANALYZER SOFTWARE

R. 13-18 2013

Emanuel Weitschek – Istituto di Analisi dei Sistemi ed Informatica del CNR, viale Manzoni 30 - 00185 Roma, Italy. Email: emanuel.weitschek@iasi.cnr.it.

Giovanni Felici – Istituto di Analisi dei Sistemi ed Informatica del CNR, viale Manzoni 30 - 00185 Roma, Italy. Email: felici@iasi.cnr.it.

Paola Bertolazzi – Istituto di Analisi dei Sistemi ed Informatica del CNR, viale Manzoni 30 - 00185 Roma, Italy. Email: bertola@iasi.cnr.it.

ISSN: 1128–3378

Collana dei Rapporti dell'Istituto di Analisi dei Sistemi ed Informatica "Antonio Ruberti", CNR

viale Manzoni 30, 00185 ROMA, Italy

tel. ++39-06-77161
fax ++39-06-7716461
email: iasi@iasi.cnr.it
URL: http://www.iasi.cnr.it

Abstract

Clustering and classification of gene expressions are common tasks in microarray analysis. We designed and implemented a software - called Microarray Logic Analyzer (MALA) - able to classify the microarray experiments and to cluster the gene expression profiles. MALA uses a supervised machine learning paradigm and is composed on the following computational steps:

- Discretization;
- Gene clustering;
- Feature selection;
- Formulas computation;
- Classification.

After the introduction of the microarray technology this technical report describes MALA methods, software modules and its deployment with different user interfaces. The software is tested successfully on two real public available gene expression profiles and on a private real microarray Alzheimer data set. MALA is able to identify logic classification rules that distinguish the different classes of the microarray experiments and to cluster similar behaving genes. In conclusion, MALA is a powerful and reliable software for microarray gene expression analysis.

1. Introduction

The modern technology, where sophisticated instruments are coupled with the massive use of computers, has made molecular biology a science where the size of the data to be gathered and analyzed poses serious computational problems. Very large data sets are ubiquitous in computational molecular biology: The European Molecular Biology Laboratory Nucleotide Sequence Database (EMBL,[2]) has almost doubled its size every year in the past ten years, and, currently, the archive comprises over 1.7 billion records covering almost 1.7 trillion base pairs of sequences. Similarly, the Protein Data Bank (PDB, [5]) has seen an exponential growth, with the number of protein structures deposited (each of which is a large data set by itself) currenty at over 50,000. An assembly task can require to reconstruct a large genomic sequence starting from hundreds of thousands of short (100 to 1000 bp) DNA fragments [29, 3]. Microarray experiments [32, 12] produce information about the expression of hundreds of thousands of genes in hundreds of individuals at once (datas set in the order of Gigabytes), and the list goes on and on.

This abundance of large bodies of biological data calls for effective methods and tools for their analysis. The data, both structured or semi-structured, have in many cases the form of two dimensional arrays, where the rows correspond to *individuals* and the columns are associated with some *features*. While in other fields (see for instance medical data) a data set contains a large number of individuals and a small set of features in the field of molecular biology this situation is reversed, and the number of individuals is small while the number of features is very large. This is mainly due to the cost of the experiments (for instance, the DNA sequencing procedure or the phasing of genotypes require a lot of time and very complex computational procedures) and to the complexity of the molecules.

These large data sets must be analyzed and interpreted to extract all relevant information they can provide, thus separating it from extra information of little practical use. *Feature Selection* and *Classification* techniques are the main tools to pursue this task. *Feature selection* techniques are meant at identifying a small subset of important data within a large data set. *Classification* techniques are designed to identify, within the analyzed data, synthetic models that are able to explain some of the relevant characteristics contained therein. The two techniques are indeed strongly related: the selection of few relevant features among the many ones available can be considered - *per se* - already a simple model of the data, and thus an application of learning; on the other hand, feature selection is always used on large bodies of data to identify those features on which to apply a classification method to identify meaningful models.

2. Microarray analysis

A microarray or DNA array is a semiconductor device, whose aim is to determine the expression level of a large set of genes with a unique parallel experiment [32, 12]. Microarrays are formed by a grid of multiple rows and columns. Each row represents a gene and each column an experimental sample. A cell of the array is associated to a probe DNA sequence, hybridized by Watson-Crick complementarity to the DNA of a target gene, eg. the mRNA sequences. mRNA sequences contains the informations for the amino acids to form a particular protein. The microarray experimental process composed of the following steps:

- the mRNA sequences are amplified
- fluorescently tagged
- poured on the array
- the array is so hybridazed
- the array is scanned with a laser that measures the quantity of fluorescent light in each cell

This measures is the expression level of the current gene that is represented in the row of the given experiment. The microarray experiments contains a large amount of data, that can be stored in form of a matrix, where each row is associated to a gene expression level and each column to an experimental sample. The set of rows is normally very larger than the set of columns. It is on average composed by more than ten thousand of rows (genes),in the size of twenty thousand. The set of columns (experimental samples) is in the size of hundred. Microarray data sets are therefore undersampled. The matrix is therefore very informative and needs to be properly analyzed to obtain the desired biological knowledge.

New advances of microarray technology lead to a large amount of gene expression data available to biological scientists and bioinformaticians. The necessity to effectively analyze the gene expression profiles of the microarray experimental samples resulted in the development of different software tools and methods. In this paper we take into consideration two particular types of microarray data analysis:

1. gene clustering

2. experiments classification

Gene clustering is the detection of gene groups that manifest similar patterns [31]. Several clustering methods can be applied to group similar genes in the microarray experiments, for a survey on clustering methods the reader could refer to [24], [40] and [28], where the author describe another common analysis on microarrays: biclustering, a technique where the genes and the experimental samples are clustered simultaneously .
The aim of experiments classification is to distinguish between two or more classes, to which the different samples belong, eg. different cell types or tumoral vs non tumoral tissues [25]. Microarrays are often used for tissue classification, especially to detect tumors. The gene expression profiles of healthy and diseased experimental samples are analyzed through the collection of their cells. Here we have a typical two class classification problem. The final goal is to individuate the genes which are related to the disease and are able to distinguish, under certain circumstances, the experimental samples. Other goals are to to classify a new samples and to compactly describe the data with an explicative model.

Currently many classification methods and softwares for microarray experiments are available. More than hundred papers have been published on this topic. For microarray experiment classification Support Vector Machines (SVM), a standard classification technique, are often used. A comparative study of common microarray experiments classification algorithms, as SVM, decision trees, boosting, is performed in [23]. The considered data sets are seven cancer microarrays. In this study boosting methods perform best, all the methods achieve a correct classification rate in the range of 70% on raw data. The authors of [23] prove also that feature selection and discretization techniques (see below for further details) improve the classification rates. Feature selection is the choice of a subset of genes that are good candidates for the classification model computation. In [23] the 50 genes with highest information gain are selected and the data is then classified with an improvement of ca. 20%.

In [4] the tumor tissue classification problems are further investigated and also gene expression profile clustering algorithms are studied. Also in this study feature selection techniques are adopted, in particular a relevance measure is computed for every gene considering its discriminating power. SVM, boosting and Nearest Neighbour classifiers are used for performing the classification task. The authors conclude that feature selection techniques are very important in order to get correct results of 90%. Other microarray classification experiments with SVM are reported in [9, 21, 19].
In [39] a nearest neighbour approach, alazy learning method, with a new distance function is investigated. Another comparison of microarray classification methods is performed in [27], taking into consideration SVMs, nearest neighbour approaches, decision trees and error correcting output codes. No method emerges as particular performing over the others. Also in this study the authors apply feature selection methods. Another comparative analysis is presented in [41], exploiting a new method, a variant of Linear Discriminant Analysis. In [35], evolutionary algorithms are used for the selection of the most relevant genes in microarray data analysis, before the real classification.

In this paper we propose a microarray gene clustering and experiments classification software: MicroArray Logic Analyzer (MALA). This software integrates an alternative clustering method and an effective classification approach to distinguish the different experimental samples.

3. Methods: MALA - MicroArray Logic Analyzer

Microarray Logic Analyzer (MALA) is a clustering and classification software, particularly engineered for microarray gene expression analysis. The aims of MALA are to cluster the microarray gene expression profiles, in order to reduce the amount of data to be analyzed, and to classify the microarray experiments. To fulfil this objective MALA uses a machine learning process based methodology, that relies on the computational steps described below. This methodology has been applied to different biological problems, like species classification with DNA Barcode sequences [8, 38, 36], Polyomaviruses identification [37] and microarray analysis [1]. For further details on the methodology we point the reader to `http://dmb.iasi.cnr.it`
The flow of MALA is composed of three main steps:

1. the optional application of discrete cluster analysis (DCA), an efficient gene expression clustering method;
2. the selection of the most relevant (clusters of) genes (feature selection);
3. the identification of the logic formulas that best characterize the microarray samples (formula extraction).

The continuous values representing the gene expressions are discretized into a limited number of intervals for each cell of the array; the obtained discrete variables are then used to select a small subset of the genes that have strong discriminating power for the considered classes; finally the logic classification formulas are extracted.

3.1. Input - output

MALA relies on the common machine learning paradigm of training and testing: the input data is divided in two disjoint sets, one for training the software, which is used to extract the model, and one for testing the accuracy of the computed model. For dividing the data in training and testing percentage split or cross validation sampling methods are supported. The MALA software accepts as input format a comma separated values (csv) file, that is easily generable with a standard spreadsheet editor, like LibreOffice Calc or Microsoft Excel, and that reflects the intrinsic structure of a microarray experiment: every row of the file contains the expression profile of a gene, every column represents an experimental sample. The heading line should list the class labels of the experimental samples. The complete syntax is reported on the MALA website at `http://dmb.iasi.cnr.it/faq.php#dmbformat`. MALA main outputs are:

- the gene clusters and its frequencies;
- the experiments classification models as logic formulas (rules in the form of ”if-then”);
- the classification rates;
- the confusion matrices.

An example of logic classification formula is ”IF A01232423 $\geq$ 0.5 then the experimental sample is CONTROL”.

3.2. Computational steps

MALA is based on the following computational steps, which have been integrated in the software:

1. Discretization
2. Gene clustering
3. Feature selection
4. Formulas computation
5. Classification

3.3. Discretization

MALA is able to deal only with binary domains, so the gene expressions have to be transformed into discrete values and to be discretized. MALA applies a transformation and converts each gene expression into a new discrete variable that is suitable for treatment into a logic framework. This step amounts in determining a set of cut points over the range of values that each variable may assume and thus define a number of intervals over which the original variable may be considered discrete (cut points). In binarized data, the new features can be viewed as binary, or logic, variables, that indicate whether the measure of one of the original real features belongs to a certain interval. MALA supports two types of discretization, that differ on the rule adopted to select the first set of intervals. The first type uses an unsupervised rules, the second a supervised.

3.3.1. Unsupervised discretization

Unsupervised discretization techniques do not take care of the class label and compute the cut points according to the information gain or to the entropy of the given feature. In these methods the user has to submit the maximum number of desired cut points. The method computes the number of samples in each interval and reduces the number of these intervals through the elimination of empty intervals and through unification of contiguous intervals that contain the same information.

To obtain an initial set of intervals for feature f_i we consider its mean μ_i and variance σ_i over the training items, and create a number of equal sized intervals symmetrical with respect to μ and proportional in size to σ_i. Once such intervals have been created, we iterate a set of steps that merge two adjacent classes if one of them is empty, if the distributions of elements in the classes is not altered (class entropy), and finally if the reduction obtained in the entropy of the feature is negligible.

For a given feature f_i, let K_i be the set of the intervals in which f_i is discretized; its entropy h_i is given by $-\sum_{k\in K_i} f_{ik} \log f_{ik}$, where $f_{lk} = p_{lk}/n$, and p_{lk} is the number of samples included in the interval k; since $h_i = 0$ if the number of intervals K_i is equal to 1, the goal is to obtain a good tradeoff between a high level of entropy and a small number of intervals. The procedure performs the following steps on the training data set:

1. For each feature f_i the mean value μ_i and the variance σ_i of the values of the feature over the items of the training set are computed;

2. N intervals around m_i are computed, so that each interval width w_i is equal to $\sqrt{\sigma}_i/N$ (such intervals are indicated with C_{ik}, for $k = 1, .., N$);

3. For each interval C_{ik}, the total number p_{ik} of samples that are included in the interval is determined, together with their distribution in the classes.

4. The N intervals are reduced on the basis of the following three criteria:

   - if an interval is empty then it is unified with one of the smaller of its adjacent intervals;
   - if in two adjacent intervals samples of one class are strongly prevalent over samples of the other classes, the two intervals are unified;
   - if one interval is poorly populated it is unified with one of the two adjacent classes if the entropy level of the feature does not fall below a given threshold.

Given the final set of intervals, a binary representation of the values of the feature is obtained by mapping the rational value of that feature into its corresponding interval, and setting the corresponding binary variable to 1.

3.3.2. Supervised Discretization

In supervised discretization algorithms the class label is considered for discretizing the input data in order to effectively direct the process into a classification dependent task. Instead of using mean and variance of the features values, when the values of a feature belonging to different classes are not overlapping, or just partially overlapping, we can use a simpler partitioning. The values are ordered and scanned in incremental way. An interval is defined as a sequence of consecutive values of the same class, once an element of a different class is found then another interval begins. Note that in this way there are not empty intervals. This alternative method performs well when it is possible to partition the values in few intervals, otherwise the reduction phase of intervals would not lead to a reduction. The worst case for this partitioning method is when the values are alternating. Let $\mathcal{F}$ be the set of features, let $f \in \mathcal{F}$ be the a feature. Let v_i be the i-th value of feature f. Every value $v_i \in f$ has a class label associated $c(f) = s$. See Figure 1 for the pseudo-code of the algorithm.

```
MALA supervised discretization Algorithm
for every numeric feature $f_i$
  $f' \leftarrow$ orderAscending($f_i$)
  $K = \emptyset$
  for every value $v_j \in f'$
    if $v_j \neq v_{i+1}$
      define $K_k = \frac{v_i + v_{i+1}}{2}$
    end if
  end for
  mergeminpop
  mergentro
end for
```

Figure 1: MALA supervised discretization algorithm.

A (mergeminpop) function merges two intervals, if one of them has an amount of population population lower than a given threshold. The (mergentro) function merges two intervals according to the entropy values: if the entropy of the interval K_i K_{i+2} is less than K_i K_{i+1} then the intervals (K_i, K_{i+1}) and (K_{i+1}, K_{i+1}) are merged in one (K_i, K_{i+2}).

3.4. Gene clustering

Microarray data sets are characterized by a large number of genes in every sample (in the range of tens of thousands); it is therefore very important to adopt methods to extract a subset of genes able to characterize the model among the exponential number of potential ones. The gene clustering step aim is to group together similar genes, whose expression or co-expression is related. MALA implements a method named Discrete Cluster Analysis (DCA) to obtain this goal.

After the discretization step an integer mapping is computed for every gene expression, that represents the interval in which an experiment falls. This integer mapping can be represented in a binary form. Two or more genes are merged into the same cluster when their binary representation over the intervals is equal. Finally, a gene for each cluster is elected as its representative. Clusters composed of a single gene may also be present and are considered as non clustered genes.

3.5. Feature selection

Feature selection is the identification of a small subset of important attributes or features in a large data set [7]. In order to obtain another substantial reduction of the genes [35], for performing the classification of the experiments, MALA applies a feature selection step. It consists in the choice of a small number of (clusters of) genes, that are candidate to distinguish the different classes of the experimental samples. MALA approaches feature selection as a combinatorial problem [11]. With binary features the problem can be mapped into a generalized Set Covering Problem (a formal definition of the FS problem (called test cover) is presented in [20]). MALA adopts a modified formulation of the test cover optimization model: the number of features to be selected is fixed in advance by the parameter β, and the level of redundancy α is maximized in the objective function:

$$\begin{aligned} \max \quad & \alpha \\ & \sum_{j=1}^{m} a_{ij} x_j - \alpha \geq 0 \quad & i = 1 \dots M \\ & \sum_{j=1}^{m} x_j \leq \beta \\ & x_j \in \{0,1\} & j = 1 \dots m, \end{aligned} \tag{1}$$

To solve the feature selection problem MALA uses an efficient heuristic named GRASP Greedy Randomized Adaptive Search Procedure) proposed in Feo and Resende [15, 16, 30, 17] and extended in [18]. An exhaustive description of the solution method and of the implementation is reported in [6] and related papers.

3.6. Formulas computation

After the output of the candidate (cluster of) genes from the GRASP algorithm MALA adopts a MinSat approach [13] for learning the logic classification formulas. The Lsquare [14, 34] method is part of MALA for computing the model of the microarray data, eg. the separating "if-then" formulas or rules. Lsquare approaches this challenge as a sequence of Minimum Cost Satisfiability Problems (MinSat), a combinatorial optimization problem that is NP-Hard. Therefore it solves the problem with an algorithm based on decomposition techniques. The reader may refer to [13, 14, 34] for a detailed description of the method and problems. MALA computes for every class of the experimental samples the logic classification formulas in Disjunctive Normal Form. The literals of the formula represent inequalities in the form of, eg. "A01232423 $\geq$ 0.5", and are conjucted in "AND" and "OR" clauses.

3.7. Classification

The evaluation of the logic formulas and the classification of the samples to the right class is performed according to the algorithm described in [38]. To summarize the process the formulas are firstly weighted with the Laplace Score [33] on the training set and then applied on the test set for performing the classification assignments. Additional cut offs of logic formulas with sub-optimal coverage are done by considering the false positive and true positive rates.

3.8. Supplementary analysis

Two additional useful microarray statistical tests have been integrated in the software MALA:

- The Pearson correlation
- The Principal Component Analysis (PCA) [26]

The widely adopted Pearson correlation analysis is available on the gene expression profiles, while the PCA can be computed both on the experiments and on the gene expression profiles. PCA and Pearson correlation may give an alternative grouping of the genes to Discrete Cluster Analysis (DCA), described above. The supplementary analysis are integrated in the graphic user interface of MALA, that is reported below.

3.9. Software engineering and releases

All the diffrent computational steps have been engineered in an integrated software named MALA, written in ANSI C and compilable on Windows, Linux and Mac OS operating systems. MALA relies on the software architectural pattern Pipes and Filters [10] for computing streams of data. A component diagram is given in figure 2. Diverse versions are released and described below.

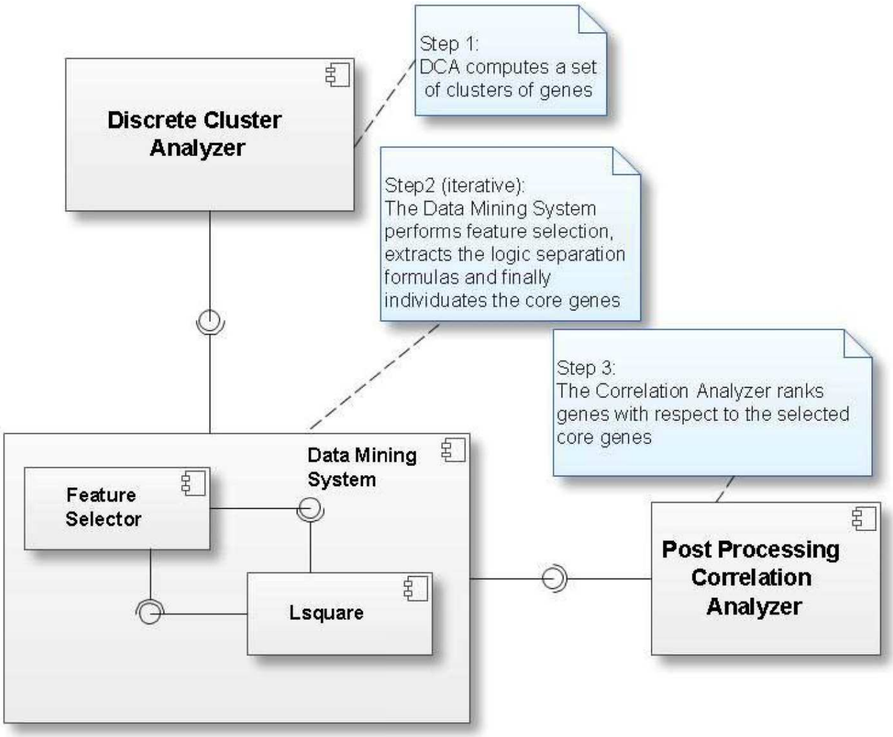

Step 1:
DCA computes a set of clusters of genes
Discrete Cluster Analyzer
Step2 (iterative):
The Data Mining System performs feature selection, extracts the logic separation formulas and finally individuates the core genes
Step 3:
The Correlation Analyzer ranks genes with respect to the selected core genes
Data Mining System
Feature Selector
Lsquare
Post Processing Correlation Analyzer

Figure 2: MALA component diagram

3.9.1. Graphic user interface

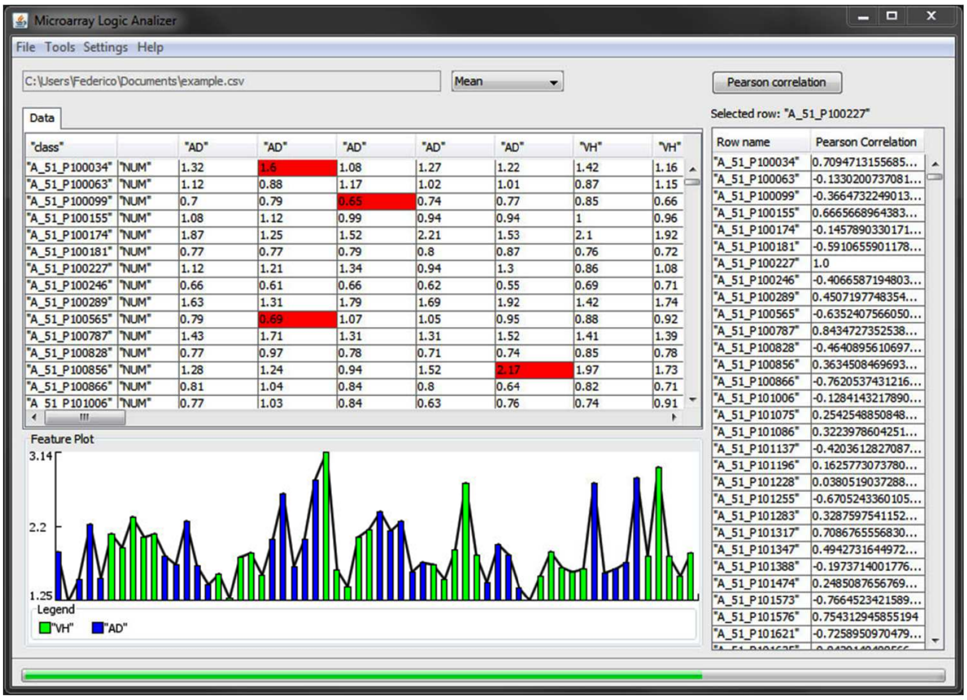
A graphic user interface, downloadable on `http://dmb.iasi.cnr.it/mala-downloads.php`, guides the user in the microarray clustering and classification process. This version uses a Java Swing framework for visualizing the data set, running the software, performing the additional analysis, viewing the clusters, the classification results and the logic separating formulas. A complete user manual is available on the MALA website. A screenshot of the offline graphic user interface is provided in figure 3.

3.9.2. Command line version

The command line version is dedicated to the users, who want to perform long experiments and batch the entire analysis process. This version is compiled and tested on Linux and Windows operating systems and available on `http://dmb.iasi.cnr.it/mala-downloads.php`. The source code is also released on the same web site for compiling MALA on alternative systems and a complete user manual is published.

3.9.3. Web service

MALA has been released also as a web service at `http://dmb.iasi.cnr.it/mala.php`. The user can upload the microarray data via an input form. After the computation, all outputs of MALA are provided



C:\Users\Federico\Documents\example.csv

Mean

"class"		"AD"	"AD"	"AD"	"AD"	"AD"	"VH"	"VH"
"A_51_P100034"	"NUM"	1.32	1.6	1.08	1.27	1.22	1.42	1.16
"A_51_P100063"	"NUM"	1.12	0.88	1.17	1.02	1.01	0.87	1.15
"A_51_P100099"	"NUM"	0.7	0.79	0.65	0.74	0.77	0.85	0.66
"A_51_P100155"	"NUM"	1.08	1.12	0.99	0.94	0.94	1	0.96
"A_51_P100174"	"NUM"	1.87	1.25	1.52	2.21	1.53	2.1	1.92
"A_51_P100181"	"NUM"	0.77	0.77	0.79	0.8	0.87	0.76	0.72
"A_51_P100227"	"NUM"	1.12	1.21	1.34	0.94	1.3	0.86	1.08
"A_51_P100246"	"NUM"	0.66	0.61	0.66	0.62	0.55	0.69	0.71
"A_51_P100289"	"NUM"	1.63	1.31	1.79	1.69	1.92	1.42	1.74
"A_51_P100565"	"NUM"	0.79	0.69	1.07	1.05	0.95	0.88	0.92
"A_51_P100787"	"NUM"	1.43	1.71	1.31	1.31	1.52	1.41	1.39
"A_51_P100828"	"NUM"	0.77	0.97	0.78	0.71	0.74	0.85	0.78
"A_51_P100856"	"NUM"	1.28	1.24	0.94	1.52	2.17	1.97	1.73
"A_51_P100866"	"NUM"	0.81	1.04	0.84	0.8	0.64	0.82	0.71
"A_51_P101006"	"NUM"	0.77	1.03	0.84	0.63	0.76	0.74	0.91

Selected row: "A_51_P100227"

Row name	Pearson Correlation
"A_51_P100034"	0.7094713155685...
"A_51_P100063"	-0.1330200737081...
"A_51_P100099"	-0.3664732249013...
"A_51_P100155"	0.6665668964383...
"A_51_P100174"	-0.1457890330171...
"A_51_P100181"	-0.5910655901178...
"A_51_P100227"	1.0
"A_51_P100246"	-0.4066587194803...
"A_51_P100289"	0.4507197748354...
"A_51_P100565"	-0.6352407566050...
"A_51_P100787"	0.8434727352538...
"A_51_P100828"	-0.4640895610697...
"A_51_P100856"	0.3634508469693...
"A_51_P100866"	-0.7620537431216...
"A_51_P101006"	-0.1284143217890...
"A_51_P101075"	0.2542548850848...
"A_51_P101086"	0.3223978604251...
"A_51_P101137"	-0.4203612827087...
"A_51_P101196"	0.1625773073780...
"A_51_P101228"	0.0380519037288...
"A_51_P101255"	-0.6705243360105...
"A_51_P101283"	0.3287597541152...
"A_51_P101317"	0.7086765556830...
"A_51_P101347"	0.4942731644972...
"A_51_P101388"	-0.1973714001776...
"A_51_P101474"	0.2485087656769...
"A_51_P101573"	-0.7664523421589...
"A_51_P101576"	0.754312945855194
"A_51_P101621"	-0.7258950970479...

Figure 3: MALA graphic offline user interface

in CSV (Comma Separated Values) format, which is easily readable by all common spreadsheet software. A compressed archive containing the computation results is sent via email to the user. The MALA web service has been released on a Linux server (Ubuntu Server distribution), using a LAMP platform (Linux Kernel 2.6.32, Apache 2.2.14, PHP 5.3.2) with a Java job queuing system that relies on a MySQL database (version 5.1.61).
A screenshot of the MALA web service is reported in figure 4.

4. Results and discussion

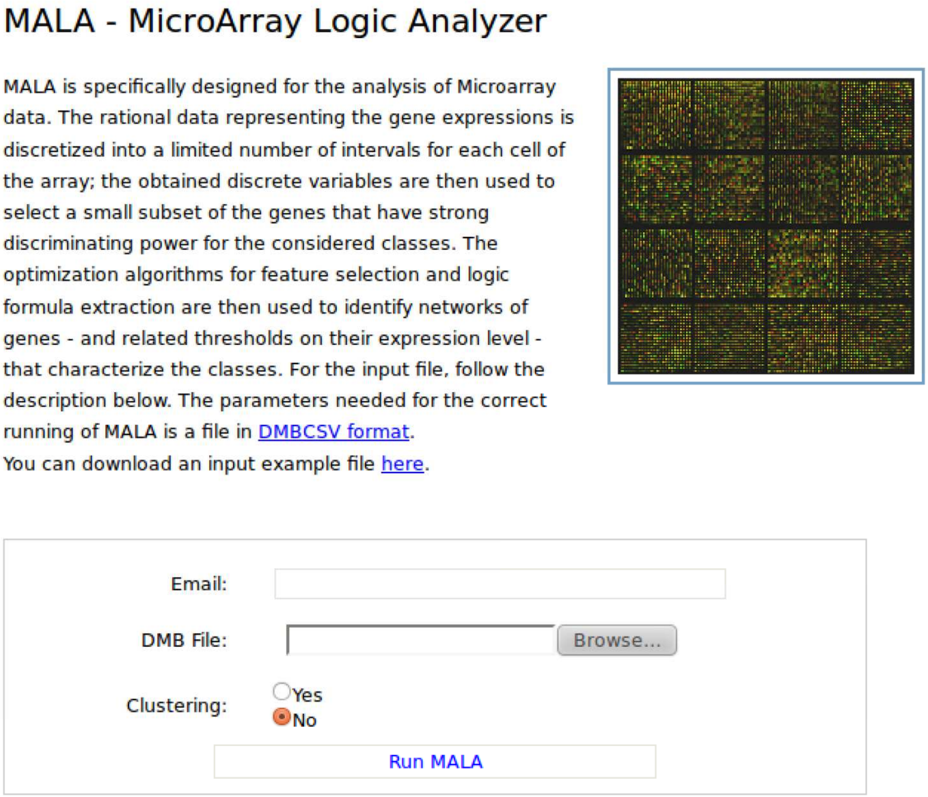
In this section the experimental results obtained in [1] to show the effectiveness and the efficacy of MALA on real microarrays gene expression profile analysis are summarized. Additional tests and comparative analysis with other classification methods have been performed on the same data set and on public available data.
In [1] MALA was used for the classification of control versus Alzheimer diseased mice, represented in early (1-3 months) and late stage (6-15 months) expression data. A small number of classification formulas was computed, encompassing mRNAs whose expression levels were able to discriminate between diseased and control mice. The purpose of the work was to discover genes whose expression or co-expression strongly characterizes the Alzheimer disease. A small number of genes capable to separate effectively between control and diseased mice was identified. The data set was composed of 119 experimental samples and 16,515 gene expression profiles and was provided from the European Brain Research Institute (EBRI). The application of the Discrete Clustering method DCA (see above) shrinked the whole gene set down to 3656 for 1-3 months and to 3615 for 6-15 months. MALA was capable to identify a few subset of genes and to compute the logic separating formulas for each class. The logic separating formulas for 1–3 and 6–15 months are reported respectively in table 1 and table 2 as examples. Every disjunctive clause reported in the table is capable to distinguish alone the two different classes of samples. The logic formulas have been validated both using a 30-fold cross validation and a holdout validation (90% train and 10% test), resulting in 99% of correct classification rate.
To reinforce the validity of the results here presented, some of the most commonly used classification

early stage	
AD	(Nudt19 < 0.76) OR
	(Arl16 ≥ 1.31) OR
	(Aph1b ≥ 0.47) OR
	(Slc15a2 ≥ 0.55) OR
	(Agpat5 ≥ 0.73) OR
	(Sox2ot < 0.58 OR Sox2ot ≥ 1.53) OR
	(2210015D19Rik ≥ 0.86) OR
	(Wdfy1 ≥ 1.37)
Control	(Nudt19 ≥ 0.76) OR
	(Arl16 < 1.31) OR
	(Aph1b < 0.47) OR
	(Slc15a2 < 0.55) OR
	(Agpat5 < 0.73) OR
	(0.58 ≥ Sox2ot AND Sox2ot < 1.53) OR
	(2210015D19Rik < 0.86) OR
	(Wdfy1 < 1.37)

Table 1: Logic formulas in early stage

6-15 months	
AD	(Slc15a2 ≥ 0.62) OR
	(Agpat5 < 0.26 OR Agpat5 ≥ 0.55) OR
	(Sox2ot ≥ 1.78) OR
	(2210015D19Rik ≥ 0.82) OR
	(Wdfy1 < 0.75 OR Wdfy1 ≥ 1.29) OR
	(D14Ertd449e < 0.33
	OR D14Ertd449e ≥ 0.52) OR
	(Tia1 < 0.17 OR Tia1 ≥ 0.49) OR
	(Txnl4 < 0.74) OR
	(1810014B01Rik < 0.71
	OR 1810014B01Rik ≥ 1.17) OR
	(Snhg3 < 0.16 OR Snhg3 ≥ 0.35) OR
	[(1.12 ≥ Actl6a AND
	Actl6a < 1.42) OR Actl6a ≥ 1.48] OR
	(Rnf25 < 0.67 OR Rnf25 ≥ 1.26)
Control	(Slc15a2 < 0.62) OR
	(0.26 ≥ Agpat5 AND Agpat5 < 0.55) OR
	(Sox2ot < 1.78) OR
	(2210015D19Rik < 0.82) OR
	(0.75 ≥ Wdfy1 AND Wdfy1 < 1.29) OR
	(0.33 ≥ D14Ertd449e AND
	D14Ertd449e < 0.52) OR
	(0.17 ≥ Tia1 AND Tia1 < 0.49) OR
	(Txnl4 ≥ 0.74) OR
	(0.71 ≥ 1810014B01Rik
	AND 1810014B01Rik < 1.17) OR
	(0.16 ≥ Snhg3 AND Snhg3 < 0.35) OR
	[(0.81 < Actl6a AND Actl6a < 1.12) OR
	(1.42 < Actl6a AND Actl6a < 1.48)] OR
	(0.67 ≥ Rnf25 AND Rnf25 < 1.26)

Table 2: Logic formulas in late stage.



MALA - MicroArray Logic Analyzer

MALA is specifically designed for the analysis of Microarray data. The rational data representing the gene expressions is discretized into a limited number of intervals for each cell of the array; the obtained discrete variables are then used to select a small subset of the genes that have strong discriminating power for the considered classes. The optimization algorithms for feature selection and logic formula extraction are then used to identify networks of genes - and related thresholds on their expression level - that characterize the classes. For the input file, follow the description below. The parameters needed for the correct running of MALA is a file in DMBCSV format.
You can download an input example file here.

Email:

DMB File: Browse...

Clustering: ○ Yes ● No

Run MALA

Figure 4: MALA web interface

algorithms were tested on the same data sets. The WEKA [22] implementations of K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Random Forest (RF) and C4.5 Classification Tree (C4.5) were adopted. Reasonable parameter tuning was performed, when necessary, for all these methods; the best settings and the correct recognition rates obtained are summarized in the following table 3. Experiments were run using a 30-fold cross validation scheme. From the results it can be seen that MALA performs at a comparable level to (slightly dominates) the other methods ; the second best is SVM, that unfortunately produces classification models whose interpretation is very difficult for human beings. As anticipated, the advantages of MALA reside in its ability to extract meaningful and compact models, in its clustering capabilities and in its availability as an integrated tool.

method	**settings**	**early stage**	**late stage**	**model**
MALA	no settings	100.0	100.0	yes
SVM	polykernel=2	96.66	100.0	no
RF	trees=100	96.66	94.91	no
C4.5	unpruned, minobj=2	98.33	98.30	yes
KNN	k=2	70.00	86.44	no

Table 3: Classification results in %

Other tests have been performed on data sets downloaded from public repositories ArrayExpress and GEO: Psoriasis and Multiple Sclerosis Diagnostic. The Psoriasis data set was composed of 54,613 gene expression profiles of 176 experimental samples (85 control and 91 diseased) and was provided from the National Psoriasis Foundation. The Multiple Sclerosis Diagnostic data set contained 22,215 gene expression profiles of 178 experimental samples (44 control and 134 diseased) and was released from the National Institute of Neurological Disorders and Stroke (NINDS). All gene expression profile values were normalized using the standard Affymetrix Expression Console software (ver 1.2), by the MAS5 algorithm. The results are reported in table 4 Also in this case MALA performs at a comparable level to (slightly dominates) the other methods. The second bests are SVM and RF, that unfortunately produce classification models that are difficult to understand for humans.

method	settings	MsDiagnostic	Psoriasis	model
MALA	no settings	94.94	100.0	yes
SVM	polykernel=2	90.45	98.86	no
RF	trees=100	91.57	98.86	no
C4.5	unpruned, minobj=2	87.08	97.16	yes
KNN	k=2	87.64	99.43	no

Table 4: Classification results in %

5. Conclusion

This paper described MALA, a clustering and classification software, particularly engineered for microarray gene expression analysis. MALA is able to cluster the gene expression profiles and to classify the microarray experimental samples. It uses a machine learning methodology based on the following steps:1) Discretization 2) Gene clustering 3) Feature selection 4) Formulas computation 5) Classification. The software is available on `http://dmb.iasi.cnr.it/mala-downloads.php` in its various releases for all common operating systems. The efficacy of MALA was showed by applying the software on a real microarray data set provided by the European Brain Research Institute (EBRI) in which CONTROL vs ALZHEIMER diseased mice were spotted on a microarray and on public available data. MALA was able to distinguish the classes present in the datasets in a precise and effective way, by computing a compact and clear model of the data: logic classification formulas in the form of "if-then rules", which can also be used to concisely describe the different classes of the data set.

Acknowledgment

The authors would like to thank Giulia Brunori, Guido Drovandi, Alessandro Giacomini, Gabriele Giammusso and Federico Russo for the contribution to the offline graphic user interface of MALA. A special thank to Ivan Arisi from EBRI for providing the data sets and for precious suggestions for MALA software features.

14.

References

[1] I. Arisi, M. D'Onofrio, R. Brandi, A. Felsani, S. Capsoni, G. Drovandi, G. Felici, E. Weitschek, P. Bertolazzi, and A. Cattaneo, "Gene expression biomarkers in the brain of a mouse model for alzheimer's disease: mining of microarray data by logic classification and feature selection," *Journal of Alzheimer's Disease*, vol. 24, no. 4, pp. 721–38, 2010.

[2] W. Baker, A. van den Broek, E. Camon, P. Hingamp, P. Sterk, G. Stoesser, and M. A. Tuli, "The embl nucleotide sequence database.," *Nucleic Acids Res*, vol. 28, pp. 19–23, Jan 2000.

[3] S. Batzoglou, D. B. Jaffe, K. Stanley, J. Butler, S. Gnerre, E. Mauceli, B. Berger, J. P. Mesirov, and E. S. Lander, "Arachne: a whole-genome shotgun assembler.," *Genome Res*, vol. 12, pp. 177–189, Jan 2002.

[4] A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini, "Tissue classification with gene expression profiles.," *J Comput Biol*, vol. 7, no. 3-4, pp. 559–583, 2000.

[5] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The Protein Data Bank," *Nucleic Acids Research*, vol. 28, pp. 235–242, January 2000.

[6] P. Bertolazzi, G. Felici, P. Festa, and G. Lancia, "Logic classification and feature selection for biomedical data," *Comput. Math. Appl.*, vol. 55, pp. 889–899, March 2008.

[7] P. Bertolazzi, G. Felici, and G. Lancia, "Application of feature selection and classification to computational molecular biology," in *Biological Data Mining* (S. L. e. J.K. Chen, ed.), pp. 257–294, Chapman & Hall, 2010.

[8] P. Bertolazzi, G. Felici, and E. Weitschek, "Learning to classify species with barcodes," *BMC Bioinformatics*, vol. 10, no. S-14, p. 7, 2009.

[9] M. P. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, M. Ares, Jr, and D. Haussler, "Knowledge-based analysis of microarray gene expression data by using support vector machines.," *Proc Natl Acad Sci U S A*, vol. 97, pp. 262–267, Jan 2000.

[10] F. Buschmann, K. Henney, and D. Schmidt, *Pattern-Oriented Software Architecture: A Pattern Language for Distributed Computing.* Volume 4. Wiley Chichester, UK, 2007.

[11] M. Charikar, V. Guruswami, R. Kumar, S. Rajagopalan, and A. Sahai, "Combinatorial feature selection problems," in *FOCS*, pp. 631–640, 2000.

[12] M. Chee, R. Yang, E. Hubbell, A. Berno, X. C. Huang, D. Stern, J. Winkler, D. J. Lockhart, M. S. Morris, and S. P. Fodor, "Accessing genetic information with high-density dna arrays.," *Science*, vol. 274, pp. 610–614, Oct 1996.

[13] G. Felici and K. Truemper, "A minsat approach for learning in logic domains," *INFORMS Journal on Computing*, vol. 13, no. 3, pp. 1–17, 2002.

[14] G. Felici and K. Truemper, *Encyclopedia of Data Warehousing and Mining, J. Wang (ed.)*, vol. 2, ch. The Lsquare System for Mining Logic Data, pp. 693–697. Idea Group Inc., 2006.

[15] T. Feo and M. Resende, "A probabilistic heuristic for a computationally difficult set covering problem," *Operations Research Letters*, vol. 8, pp. 67–71, 1989.

[16] T. Feo and M. Resende, "Greedy randomized adaptive search procedures," *Journal of Global Optimization*, vol. 6, pp. 109–133, 1995.

[17] P. Festa and M. Resende, *C.C. Ribeiro, P. Hansen (Eds.), Essays and Surveys on Metaheuristics*, ch. Grasp: An annotated bibliography, pp. 325–367. Kluwer Academic Publishers, 2002.

[18] P. Festa and M. Resende, “An annotated bibliography of grasp – part i: Algorithms,” *International Transactions in Operational Research*, vol. 16, no. 1, pp. 1–24, 2009.

[19] 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*, vol. 16, pp. 906–914, Oct 2000.

[20] M. Garey and D. Johnson, *Computer and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, 1979.

[21] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Machine Learning*, vol. 46, no. 1-3, pp. 389–422, 2002.

[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The weka data mining software: an update,” *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, Nov. 2009.

[23] H. Hu, J. Li, A. W. Plank, H. Wang, and G. Daggard, “A comparative study of classification methods for microarray data analysis,” in *AusDM*, pp. 33–37, 2006.

[24] D. Jiang, C. Tang, and A. Zhang, “Cluster analysis for gene expression data: a survey,” *Knowledge and Data Engineering, IEEE Transactions on*, vol. 16, no. 11, pp. 1370 – 1386, 2004.

[25] T. Jirapech-Umpai and S. Aitken, “Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes,” *BMC Bioinformatics*, vol. 148, 2005.

[26] I. Jolliffe, *Principal component analysis.* Springer series in statistics, Springer-Verlag, 2002.

[27] T. Li, C. Zhang, and M. Ogihara, “A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression.,” *Bioinformatics*, vol. 20, pp. 2429–2437, Oct 2004.

[28] S. Madeira and A. Oliveira, “Biclustering algorithms for biological data analysis: a survey,” *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. 1, no. 1, pp. 24 –45, 2004.

[29] E. W. Myers, G. G. Sutton, A. L. Delcher, I. M. Dew, D. P. Fasulo, M. J. Flanigan, S. A. Kravitz, C. M. Mobarry, K. H. Reinert, K. A. Remington, E. L. Anson, R. A. Bolanos, H. H. Chou, C. M. Jordan, A. L. Halpern, S. Lonardi, E. M. Beasley, R. C. Brandon, L. Chen, P. J. Dunn, Z. Lai, Y. Liang, D. R. Nusskern, M. Zhan, Q. Zhang, X. Zheng, G. M. Rubin, M. D. Adams, and J. C. Venter, “A whole-genome assembly of drosophila.,” *Science*, vol. 287, pp. 2196–2204, Mar 2000.

[30] M. Resende and C. Ribeiro, *F. Glover, G. Kochenberger (Eds.), State-of-the-Art Handbook of Metaheuristics*, ch. Greedy randomized adaptive search procedures, pp. 219–249. Kluwer, 2002.

[31] L. Romdhane, H. Shili, and B. Ayeb, “Mining microarray gene expression data with unsupervised possibilistic clustering and proximity graphs,” *Applied Intelligence*, vol. 33, pp. 220–231, 2010.

[32] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown, “Quantitative monitoring of gene expression patterns with a complementary dna microarray.,” *Science*, vol. 270, pp. 467–470, Oct 1995.

[33] P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining.* Addison Wesley, 2005.

[34] K. Truemper, *Design of Logic-Based Intelligent Systems.* Wiley-Interscience, 2004.

[35] T. J. Umpai and S. Aitken, “Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes,” *BMC Bioinformatics*, vol. 6, no. 1, pp. 1–11, 2005.

[36] R. van Velzen, E. Weitschek, G. Felici, and F. T. Bakker, “Dna barcoding of recently diverged species: Relative performance of matching methods,” *PLoS ONE*, vol. 7, p. e30490, 01 2012.

[37] E. Weitschek, A. Lo Presti, G. Drovandi, G. Felici, M. Ciccozzi, M. Ciotti, and P. Bertolazzi, “Human polyomaviruses identification by logic mining techniques,” *BMC Virology Journal*, vol. 58, no. 9, 2012.

[38] E. Weitschek, R. VanVelzen, and G. Felici, “Species classification using dna barcode sequences: A comparative analysis,” Tech. Rep. 11-07, IASI CNR, 2011.

[39] H. Xiong and X.-w. Chen, “Kernel-based distance metric learning for microarray data classification.,” *BMC Bioinformatics*, vol. 7, p. 299, 2006.

[40] R. Xu and D. C. W. II, “Survey of clustering algorithms,” *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.

[41] J. Ye, T. Li, T. Xiong, and R. Janardan, “Using uncorrelated discriminant analysis for tissue classification with gene expression data,” vol. 1, no. 4, pp. 181–190, 2004.