A cDNA Microarray Gene Expression Data Classifier for Clinical Diagnostics Based on Graph Theory

Alfredo Benso, Stefano Di Carlo, and Gianfranco Politano

Abstract—Despite great advances in discovering cancer molecular profiles, the proper application of microarray technology to routine clinical diagnostics is still a challenge. Current practices in the classification of microarrays' data show two main limitations: the reliability of the training data sets used to build the classifiers, and the classifiers' performances, especially when the sample to be classified does not belong to any of the available classes. In this case, state-of-the-art algorithms usually produce a high rate of false positives that, in real diagnostic applications, are unacceptable. To address this problem, this paper presents a new cDNA microarray data classification algorithm based on graph theory and is able to overcome most of the limitations of known classification methodologies. The classifier works by analyzing gene expression data organized in an innovative data structure based on graphs, where vertices correspond to genes and edges to gene expression relationships. To demonstrate the novelty of the proposed approach, the authors present an experimental performance comparison between the proposed classifier and several state-of-the-art classification algorithms.

Index Terms—Microarray, gene expression, classification, clinical diagnostics, graph theory.

1 Introduction

DNA microarrays are one of the fastest growing technologies for genetic research. They are small solid supports, e.g., membranes or glass slides, on which sequences of DNA are fixed in an orderly arrangement. Tens of thousands of DNA probes can be attached to a single slide and used to analyze and measure the activity of genes. Scientists are using DNA microarrays to investigate several phenomena (e.g., cancer, pest control, etc.) by measuring changes in gene expression and thereby learning how cells respond to a disease or to a particular treatment [1], [2]. Even if microarrays represent a powerful source of biological information, using gene expression data to classify diseases on a molecular level for clinical diagnostic remains a challenging research problem. It involves assessing gene expression levels from different experiments, determining genes whose expression is relevant (*feature extraction*), and then, applying accurate and readily interpretable classification rules providing biological insight of the target phenomenon (*classification*) [3].

Classifying microarray data poses several challenges to typical machine learning methods. In particular, microarray classification faces the *"small N, large P"* problem of statistical learning, where the number P of variables (gene expressions) is typically much larger than the number N of available samples. This disparity impacts the major aspects of the classifier design: the classification rule, the error estimation, and the feature selection. Moreover, when considering clinical diagnostic, one of the main problems of traditional machine learning techniques concerns the ability of properly detecting false positives, i.e., samples erroneously assigned to a class even if they do not belong to the class library used to train the classifier. This misbehavior is clearly unacceptable since it would very likely lead to a misdiagnosis.

This paper tries to leverage these problems by presenting a new classification algorithm based on a graph-based data structure, called *Gene Expression Graph* (GEG), used to represent gene expression data. GEGs clearly express relationships among expressed and silenced genes in different conditions (e.g., healthy versus diseased). They are, therefore, specifically suited for the analysis of complementary DNA (cDNA) microarrays that provide on a single support information for both healthy and diseased specimens. The concept of using a graph-based data structure to represent gene expression data, first published by the authors in [4] and [5], allows to build a classifier based on a topological comparison between graphs representing known classes and graphs representing samples to analyze.

One of the main contributions of the classifier stems in the ability of combining in a single algorithm high accuracy in the classification process together with the ability of detecting samples not belonging to any of the trained classes, thus drastically reducing the number of false positive classification outcomes. To validate the efficiency of the proposed approach, the paper presents an experimental comparison between the GEG-based classifier and several generic state-of-the-art multiclass and one-class classification methods on a set of cDNA microarray experiments for 15 well-known and documented diseases. Experimental results show that the GEG-based classifier is

• *The authors are with the Control and Computer Engineering Department, Politecnico di Torino, Corso Duca degli Abruzzi 24, I-10129, Torino, Italy. E-mail: {alfredo.benso, stefano.dicarlo, gianfranco.politano}@polito.it.*

Manuscript received 1 July 2009; revised 24 Mar. 2010; accepted 24 May 2010; published online 9 Sept. 2010.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-2009-07-0114. Digital Object Identifier no. 10.1109/TCBB.2010.90.

able to reach the same performances reached by multiclass classifiers when dealing with samples belonging to the considered class library, while it outperforms one-class classifiers in the ability of detecting samples not belonging to any of the trained classes.

The paper is organized as follows: Section 2 describes the proposed classification approach, and Section 3 proposes an overview of existing classification methods in order to highlight the main differences with the proposed approach. Section 4 presents the experimental results, and Section 5 concludes the paper suggesting future developments of this work.

2 Methodology

Classifying diseases based on DNA Microarray gene expression information usually requires:

1. signal preprocessing of raw microarray scans,
2. data modeling,
3. prediction (e.g., classification), and
4. validation.

The signal preprocessing stage elaborates the raw image obtained from the scanning process of a microarray (*sample*) in order to calculate the *expression level* of each DNA probe. It compensates for any kind of acquisition errors, hardware damages (scratches, dust, etc.), and procedural issues (stains, drops, spots out of focus, etc). While this step is out of the scope of this paper, the next sections will focus on the problem of efficiently modeling microarray data, and on the definition of an efficient prediction algorithm for classification of diseases.

2.1 Data Modeling

The result of the scanning process and signal processing of a microarray containing probes (spots) for P different genes is a *gene expression profile*

$$\vec{s} = (expr_1, expr_2, \ldots, expr_P), \tag{1}$$

with $expr_j$ representing the *gene expression level* of the jth gene (g_j) of the sample $(j \in [1, P] \subset \mathbb{N})$.

Depending on the microarray technology $expr_j$ may identify an absolute expression level or a set of expression levels measured in different conditions. cDNA microarrays, the main target of this paper, provide for each spot two expression levels using two fluorescence intensities: the first labeled Cy5 producing a red fluorescence associated with a diseased condition, and the second labeled Cy3 producing a green fluorescence associated with a healthy condition. We can, therefore, formally define $expr_j$ as an index function

$$expr_j : \{Cy5, Cy3\} \rightarrow \mathbb{R}, \tag{2}$$

and denote with $\vec{s}[j][Cy5]$ and $\vec{s}[j][Cy3]$ the two expression levels of gene g_j in sample $\vec{s}$.

Gene expression profiles obtained from N samples are usually organized in the form of a matrix called Gene Expression Matrix (GEM) [6], where the ith row $(i \in [1, N] \subset \mathbb{N})$ represents the gene expression profile of the ith sample of the considered set. The GEM obtained from the scanning process of a set of microarrays is a raw data set containing noise, missing values, and systematic variations arising from the experimental procedure. Techniques, such as the one proposed in [7], are widely applied to mitigate the effect of these problems and to improve the overall data quality. GEMs are a data model widely used to build classification and prediction algorithms. In this paper, we consider a multiclass classification problem. A classifier or predictor $\mathcal{C}$ for K classes, each one representing a disease, is a map from the space $\mathcal{S}$ of all possible gene expression profiles into a set Y of K classes

$$\mathcal{C} : \mathcal{S} \rightarrow Y = \{y_1, y_2, \ldots, y_K\}, \tag{3}$$

built over a training set

$$Tr = \{t_1 = (\vec{s_1}, \mathcal{C}(\vec{s_1})), \ldots, t_T = (\vec{s_T}, \mathcal{C}(\vec{s_T}))\}, \tag{4}$$

of T previously labeled samples. Since classification methods rely on gene expression profiles to perform predictions, several preprocessing steps, such as between-microarray normalization, are usually applied to make quantitative comparison of two or more microarrays possible [8], [9]. Each of these steps may strongly impact the efficiency and accuracy of the classifier. Moreover, they require rebuilding the overall data model every time new samples have to be analyzed, or added to the training set to enhance the learning capability of the classifier. This operation is computationally expensive, and might not be affordable if performance is one of the target requirements.

To cope with these problems, we propose a new graph-based data model for groups of gene expression profiles, built on raw gene expression measures. The model is constructed in order to allow efficient classification and to avoid the influence of preprocessing steps on the prediction process.

To better explain the proposed approach, let us split the training set Tr into K subsets

$$Tr = \{Tr_1, Tr_2, \ldots, Tr_K\}, \tag{5}$$

where each subset $Tr_i = \{(\vec{s_x}, \mathcal{C}(\vec{s_x})) \in Tr \mid \mathcal{C}(\vec{s_x}) = y_i\}$ characterizes a separate class y_i, and, therefore, groups samples of individuals affected by the same disease.

Each subset Tr_i can be modeled by a nonoriented weighted graph (Gene Expression Graph) $GEG_i = (V_i, E_i)$ where

- each vertex $v_x \in V_i$, with $x \in [1, P] \subset \mathbb{N}$ represents gene g_x of samples belonging to Tr_i, labeled using its UnigeneID [10]. Only vertexes representing *relevant* genes are included in the graph (the concept of *gene relevance* will be discussed in Section 2.1.1).
- each edge $(u, v) \in E_i \subseteq V_i \times V_i$ connects pairs of vertexes representing genes that are corelevant, i.e., concurrently relevant, within a single sample $\vec{s_x} \mid (\vec{s_x}, \mathcal{C}(\vec{s_x})) \in Tr_i$. It, therefore, models relationships among relevant genes of a sample. If n genes are corelevant in the same sample, each corresponding vertex will be connected with an edge to the remaining $n-1$ ones, thus creating a clique. This structural property is very important to identify features connected to each training subset Tr_i, and thus to construct an efficient classifier. This consideration is based on the hypothesis that, when

considering a statistically significant number of experiments, genes that are corelevant within the same sample are likely to have a biological meaning in characterizing the target disease (e.g., [11]).

- the weight $w_{u,v}$ of each edge $(u,v) \in E_i$ corresponds to the number of times genes u and v are corelevant in the same sample over the set of samples composing Tr_i. In a graph built over a single experiment, each edge will be weighted as 1. Adding additional microarrays will modify the graph by introducing additional edges and/or by modifying the weight of existing ones.

2.1.1 Relevant Genes Selection

The identification of *relevant* genes is a key factor to build gene expression graphs. Depending on the experimental setup and on the specific microarray technology, different strategies can be exploited. In [4], we adopted the *absolute* gene expression level of diseased tissues. This approach presents two drawbacks: 1) it requires the identification of a threshold representing a Boolean cutoff between relevant and not-relevant genes, and 2) it does not allow to model both expressed and silenced genes.

Several studies on cDNA microarrays have shown that it is possible to identify relevant genes for a target disease by searching for spots that are differentially expressed between healthy and diseased conditions [12]. The predominance of the Cy3 or Cy5 component of a spot indicates the abundance of the corresponding DNA sequence, allowing us to introduce the concept of overexpressed or silenced genes. On the other hand, equal intensities indicate no peculiar information for the corresponding gene. Based on this concept, it is possible to define different metrics to evaluate differential expressions and to identify relevant genes for constructing GEGs.

In [5], we considered the use of a *linear difference* between diseased and healthy gene expressions. Despite greatly improving the data model quality w.r.t. [4] and allowing a significant noise reduction, this process suffered again from the need of defining a differential expression threshold to distinguish between relevant and nonrelevant genes. Being linear differences of gene expressions of a single sample not heavy-tailed distributed, it was impossible to define the threshold based on a strong mathematical model. Moreover, considering genes with equal differential expression, it was impossible to discriminate between genes with high Cy5 and Cy3 components from genes with low Cy5 and Cy3 components. Loosing this information may reduce the efficiency of the classification process.

Several publications on microarray data suggest to use the (binary) logarithm of the ratio $Cy5/Cy3$ (log-ratio) to measure differential expression of genes. The use of the log-ratio seems to be a better solution to identify relevant genes for GEGs. It takes into account the absolute intensity of the two channels, thus overcoming one of the problems of the linear difference. Moreover, log-ratios of a microarray tend to be Normally distributed [13], thus allowing to build a more rigorous mathematical model to better identify relevant genes. Relevant and nonrelevant genes of a sample can, therefore, be selected considering whether the corresponding log-ratio is null (nonrelevant gene) or not (relevant gene). This simple rule can be further refined by considering the sign of the log-ratio. Genes that exhibit a positive log-ratio are upregulated in the diseased condition (Cy5 component) compared to the healthy one (Cy3) identifying *overexpressed* relevant genes. On the other hand, genes with a negative log-ratio are downregulated in the diseased condition w.r.t. the healthy one, identifying *silenced* relevant genes.

When applying this rule, one has to consider that experimental conditions may introduce systematic biases on the experimental data able to shift or scale expression levels of a sample. To compensate these problems and to allow comparison of different samples, within-microarray normalization should be applied. Among the different methods proposed in literature, we applied here the standard score (z score) normalization [14]. The standard score transformation expresses log-ratios for individual genes of a sample as a unit of the standard deviation from the normalized mean of zero

$$z(Cy5, Cy3, \mu, \sigma) = \frac{log_2 \frac{Cy5}{Cy3} - \mu}{\sigma}, \tag{6}$$

where μ and σ denote the mean and the standard deviation of log-ratios of all genes within the considered sample. Correction is done before sample-to-sample comparison and is, therefore, comparison independent.

Finally, considering a sharp cutoff between overexpressed and silenced genes limits the ability of identifying nonrelevant genes. In fact, the amount of genes that will exhibit a perfectly null standard score will be usually really small. A threshold ε could be, therefore, introduced to enlarge the nonrelevance area. Different approaches can be considered to define this threshold. It can be connected to the intrinsic error introduced when measuring expression levels of genes, or it can be defined based on well-established methods to identify differentially expressed genes, such as fold-change or t-test [15]. Even if this approach still requires the introduction of a threshold, we will show that its influence on the accuracy of the proposed classifier is negligible (see Section 4).

In a formal way, by indicating with $ER \subset \mathbb{R}$ (Expression Range) the full scale range of Cy3 and Cy5 for the target technology, the relevance can be defined as a function that maps these two components into one of three possible values indicating: (1) overexpressed relevant genes, (-1) silenced relevant genes, and (0) nonrelevant genes

$$Rel_{\varepsilon,\mu,\sigma} : ER \times ER \rightarrow \{0, 1, -1\}, \tag{7}$$

$$Rel_{\varepsilon,\mu,\sigma}(Cy5, Cy3) \rightarrow \begin{cases} 1 & z(Cy5, Cy3, \mu, \sigma) > \varepsilon, \\ 0 & -\varepsilon \leq z(Cy5, Cy3, \mu, \sigma) \leq \varepsilon, \\ -1 & z(Cy5, Cy3, \mu, \sigma) < \varepsilon. \end{cases} \tag{8}$$

Based on this definition, each node $v_x \in V_i$ of GEG_i associated with gene g_x can be additionally labeled with a so called *Cumulative Relevance Count*

$$CRC_x = \sum_{\forall j | \left(\vec{s}_j, \mathcal{C}(\vec{s}_j)\right) \in Tr_i} Rel_{\varepsilon,\mu,\sigma}\left(\vec{s}_j[x][Cy5], \vec{s}_j[x][Cy3]\right). \tag{9}$$

A node with positive *CRC* identifies a gene overexpressed in the majority of the samples, while a node

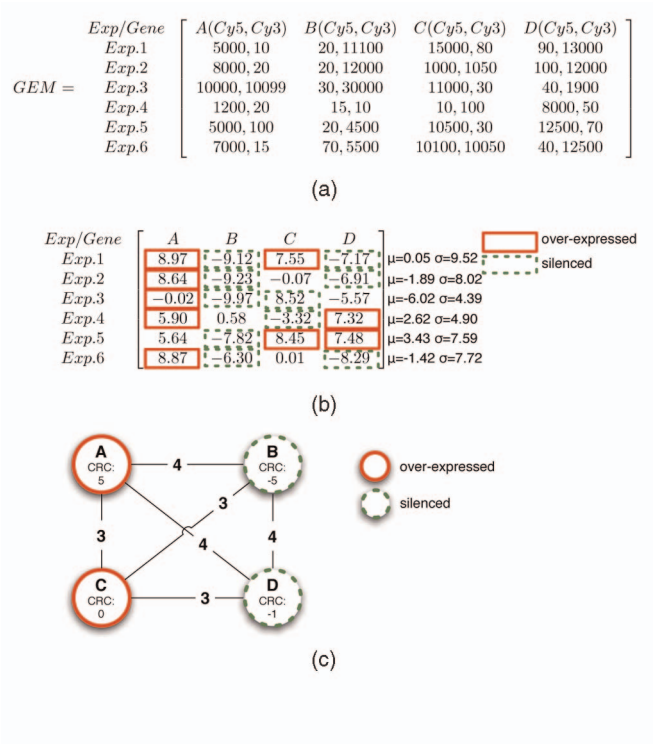

Fig. 1. GEG construction example starting from the initial set of gene expression profiles to the final graph construction. (a) Initial training set expression levels represented as a gene expression matrix. (b) Log-ratios matrix and relevance with $\varepsilon = 1$. (c) Gene expression graphs. Silenced nodes indicate nodes with negative CRC, i.e., nodes silenced in the majority of the samples, while overexpressed node represent nodes with positive CRC, i.e., nodes overexpressed in the majority of the samples.

with negative *CRC* identifies a gene silenced in the majority of the samples.

In a similar way, the weight $w_{u,v}$ of each edge $(u, v) \in E_i$ of GEG_i can be formally expressed as follows:

$$w_{u,v} = \sum_{\forall j | (\vec{s}_j, \mathcal{C}(\vec{s}_j)) \in Tr_i} \left(\left| Rel_{\varepsilon,\mu,\sigma}(\vec{s}_j[u][Cy5], \vec{s}_j[u][Cy3]) \right| \wedge \left| Rel_{\varepsilon,\mu,\sigma}(\vec{s}_j[v][Cy5], \vec{s}_j[v][Cy3]) \right| \right), \quad (10)$$

where $\wedge$ denotes the logical *and* operator returning 1, if both its operands are equal to 1, while $|\cdot|$ denotes the absolute value of the related relevance value. Each sample will, therefore, provide a unitary contribution to $w_{u,v}$ if both g_u and g_v will exhibit a relevance (in absolute value) equal to 1.

Fig. 1 shows an example of GEG construction from a set Tr of six samples (Fig. 1a). Each sample includes four genes and, for each gene, the Cy5 and Cy3 components are provided. Fig. 1b shows the log-ratio calculated for each gene in each sample and the indication of overexpressed relevant genes, silenced relevant genes, and nonrelevant genes. Relevant genes are identified according to (8) with a threshold $\varepsilon = 1$ (two-fold [15]). Starting from these values, Fig. 1c shows the corresponding GEG where each vertex corresponds to a gene that is relevant in at least one experiment. To give an example of how to compute the CRC for each vertex, and the weight of each arc, let us look in more details at vertexes C and D. Looking at the log-ratio table, one can see that gene C is overexpressed in two experiments (Exps. 1 and 5), silenced in two experiments (Exps. 3 and 4), and nonrelevant in two experiments (Exps. 2 and 6). The CRC of node C in the GEG is, therefore, $CRC_C = 2 - 2 = 0$. Gene D, instead, is silenced in three experiments, and overexpressed in two experiments; its CRC is therefore -1. To compute the weight of the edge (C, D), it is enough to count the number of experiments in which both genes are relevant (this time without taking into account the sign). They are Experiments 1, 4, and 5; the weight $w_{C,D}$ is, therefore, 3.

If new samples become available from new experiments referring to the same pathology, the related information can be easily added to the corresponding GEG without any additional memory requirement. GEGs' memory occupation is, in fact, determined by the number of considered genes, only, and independent of the number of experiments in the data set.

2.2 Classification

Gene Expression Graphs represent an excellent data structure for building efficient classifiers. The classifier presented in this paper works by structurally comparing pairs of GEGs: one representing a given pathology (GEG_{pat}), built from a corresponding training set Tr_{pat} (4), and the other representing the sample $\vec{s}$ to classify (GEG_s). This comparison measures how much GEG_s is similar (or can be overlapped) to GEG_{pat} in terms of overexpressed/silenced genes (CRC of vertexes) and relationships among gene expressions (weight of edges). The result of this operation is a *proximity score* ($Ps \in [-1, 1] \subset \mathbb{R}$), computed according to (11), measuring the similarity between the two graphs

$$Ps(GEG_{pat}, GEG_s) = \frac{SMS(GEG_{pat}, GEG_s)}{MMS(GEG_{pat})}. \quad (11)$$

Sample matching score (*SMS*) analyzes the similarity of GEG_{pat} and GEG_s considering those vertexes (genes) appearing in both graphs, only. *SMS* is computed as follows:

$$SMS(GEG_{pat}, GEG_s) = \sum_{\forall (i,j) \in E_s \cap E_{pat}} \left(Z_i \cdot w_{i,j} \cdot \frac{|Z_i|}{|Z_i| + |Z_j|} \right) + \left(Z_j \cdot w_{i,j} \cdot \frac{|Z_j|}{|Z_i| + |Z_j|} \right), \quad (12)$$

where (i, j) are edges appearing in both GEG_s and GEG_{pat}, while the term Z_x (Z-term) for vertex v_x is computed as follows:

$$Z_x = CRC_{x_{pat}} \cdot CRC_{x_s}. \quad (13)$$

By construction, each vertex v_x of a GEG has $CRC_x < 0$ if g_x is silenced in the majority of the samples of its training set, $CRC_x = 0$ if g_x is actually not relevant in its training set, or $CRC_x > 0$ if g_x is overexpressed in the majority of the samples of its training set. Z_x may, therefore, assume the following values:

- $Z_x > 0$: if g_x is silenced/overexpressed in both GEG_s and GEG_{pat};
- $Z_x < 0$: if g_x is silenced in GEG_s and overexpressed in GEG_{pat}, or viceversa;
- $Z_x = 0$: if g_x is not relevant either in GEG_s, or in GEG_{pat}.

The purpose of this term is to quantify to what extent the expression of g_x in $\vec{s}$ is "similar" to the expression of the same gene in Tr_{pat}. The more genes have a positive Z-term, the higher will be the similarity, and therefore the SMS of the sample w.r.t. the considered GEG_{pat}. The two terms of (12) are the Z-term of the two genes connected by the considered edge $((i,j))$, each one multiplied by a portion of the weight of the edge. This portion is computed as the percentage of the Z-term of the gene over the total Z-term of the pair.

Maximum matching score (MMS) is the maximum SMS that would be obtained with all genes in GEG_s perfectly matching all genes in GEG_{pat}, with Z-term of each gene always positive. It can, therefore, be computed as follows:

$$MMS(GEG_{pat}) = \sum_{\forall (i,j) \in E_{pat}} \left(w_{i,j} \cdot \frac{CRC_i^2 + CRC_j^2}{|CRC_i| + |CRC_j|} \right). \quad (14)$$

The proximity score is used as a measure of $\vec{s}$ to belong to the class (disease) characterized by GEG_{pat}. Positive values indicate the existence of similarities between the two graphs, with higher scores indicating higher similarity between the sample and the class. Negative values indicate opposite structural information between the two graphs, thus identifying high dissimilarity between the sample and the class. Considering the multiclass classification problem of (3) and given a set of K graphs, $GEG_1, GEG_2, \ldots, GEG_K$, with GEG_i built using the subset $Tr_i \subset Tr$ (5) and characterizing a given disease, we can compute a proximity vector for a sample $\vec{s}$ as follows:

$$P\vec{V}_s = (Ps(GEG_1, GEG_s), \ldots, Ps(GEG_K, GEG_s)), \quad (15)$$

where $P\vec{V}_s[i]$, with $i \in [1, K]$ represents the proximity score of sample $\vec{s}$ with class y_i, i.e., the measure of the affinity of $\vec{s}$ with the class y_i.

It is important to highlight that the computation of each proximity score composing $P\vec{V}_s$ is independent of the number of classes considered in the problem, making it an absolute indicator of the similarity of a sample with a given class. This is in contrast with several state-of-the-art classifiers (see Section 3), where proximity measures often depend on the number of classes and need to be interpreted in relation with the measure of all the considered classes. This forces to completely rebuild the prediction model every time new classes are included in the classification process.

2.2.1 Classifier's Decision Rule

Given the proximity vector of a sample $\vec{s}$, the definition of the classifier $\mathcal{C}$ (3) passes through the definition of a decision rule able to predict the correct class. In statistical classification, a decision rule uses information from a sample to decide between two or more hypotheses. State-of-the-art classifiers (see Section 3) usually apply the maximum proximity rule to identify the predicted class. In our case, this would mean choosing, as best prediction for a sample $\vec{s}$, the class with the highest Ps

$$\mathcal{C}(\vec{s}) = y_{argmax_i}\left(P\vec{V}_s[i] \mid \forall i \in [1, K] \right). \quad (16)$$

This rule is really effective when working on samples that are known to belong to the available class library. Nevertheless, when performing predictions for clinical diagnostic, two situations should be considered: 1) the sample actually belongs to one of the classes the classifier has been trained for (classifiable sample), or 2) the sample does not belong to any class (out-of-class sample) because it is either a healthy sample or a sample showing a disease not considered when the classifier was trained. The classification problem should, therefore, be extended by introducing an *out-of-class* (y_{ooc}) condition as follows:

$$\mathcal{C} : \mathcal{S} \rightarrow (Y = \{y_1, y_2, \ldots, y_K\}) \cup (y_{ooc} = (\overline{y_1} \wedge \overline{y_2} \wedge \cdots \wedge \overline{y_K})), \quad (17)$$

where the notation $\overline{y_i}$ stands for "not in class y_i" and $\wedge$ denotes the logical *and*.

Given this new classification problem, samples can be grouped based on the result of the classification into four groups:

- *True Positives:* classifiable samples classified in one of the available classes $y_i \in Y$. This includes samples of class $y_i \in Y$ classified in the correct class (*matches*), and samples of class $y_i \in Y$ classified in the wrong class $y_j \in Y$, with $y_j \neq y_i$ (*mismatches*).
- *True Negatives:* out-of-class samples correctly classified as y_{ooc},
- *False Positives:* out-of-class samples erroneously classified as $y_i \in Y$, and
- *False Negatives:* samples of class $y_i \in Y$ erroneously classified y_{ooc}.

These definitions will be used in the remaining parts of the paper to analyze the capability of the classifier to both identify the correct class for classifiable samples (matches versus mismatches), and distinguish between classifiable and out-of-class samples. In particular, failing to identify out-of-class samples creates a very high rate of false positives, dramatically reducing the applicability of the classifier to a real diagnostic scenario, where both false positives and false negatives have to be lowered as much as possible, if not completely removed.

The maximum probability rule proposed in (16) fails when trying to identify out-of-class samples. To overcome this problem, we propose an improved decision rule based on the analysis of how proximity scores distribute between classifiable and out-of class samples. Let us consider the training set Tr used to train the classifier. A set of cross-validation experiments to analyze how proximity scores distribute between classifiable and out-of-class samples can be setup as follows: K subsets of experiments are generated by removing one of the available classes $y_i (i \in [1, K])$ from the training library and using the corresponding samples Tr_i as out-of-class samples to classify. For each subset, several folds can be generated by removing k samples from the training sets of each class of the considered library and k samples from the out-of-class set. Each fold will therefore contain a test-set of $k \cdot (K-1)$ classifiable samples, and k out-of-class samples. The probability density function of the proximity scores for the given classification problem can be finally estimated by performing a kernel density estimation on the obtained results [16]. Fig. 2 shows the kernel density estimate of the proximity scores probability density function for the data set introduced in Section 4.1 to

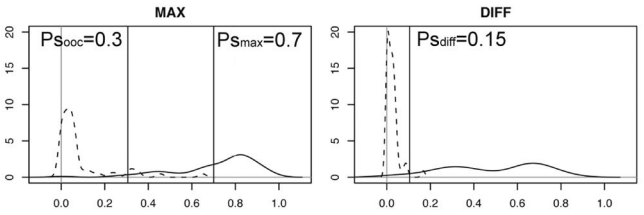

Fig. 2. Kernel density estimate of proximity scores probability density function for the GEG-based classifier. Estimation is performed using a Gaussian kernel.

validate our method, using a gaussian kernel. MAX shows the distribution of the highest proximity score of each sample for all classifiable samples (true positives—solid line), and all out-of-class samples (false positives—dotted line), while DIFF shows the difference between the values of the two highest Ps for each sample, again for both classifiable and out-of-class samples.

Looking at Fig. 2, we propose to identify three distinct areas: 1) the maximum proximity area, 2) the decision area, and 3) the out-of-class area, delimited by two thresholds Ps_{max} and $Ps_{ooc}(Ps_{max} > Ps_{ooc})$. Based on this partitioning, the following decision rules (Fig. 3) can be exploited to solve the classification problem of (17):

- R1 (maximize true negatives): to be able to predict a class for a sample $\vec{s}$ at least one class y_i $(i \in [1, K])$ should exhibit a proximity score $P\vec{V}_s[i]$ higher Ps_{ooc}. If this condition is not satisfied (out-of-class area), $\vec{s}$ is classified as y_{ooc}.
- R2 (maximize true positives): if at least one class with proximity scores higher than Ps_{max} do exist (maximum probability area), the class with maximum proximity score is predicted. This gives maximum confidence to predictions with high score.
- R3: in case neither R1 nor R2 is satisfied, the prediction should be identified among those classes with $Ps \in [Ps_{ooc}, Ps_{max}]$ (decision area). In this case, a prediction is provided if the two top ranked proximity scores differ at least of a minimum value Ps_{diff} allowing to distinguish between the two classes. In fact, this rule applies every time the prediction is not certain, i.e., low proximity score. It avoids to provide a result if the distinction between two classes is not sufficient to take a clear decision.
- R4: whenever the first three rules cannot be applied the prediction is considered uncertain and the classifier does not produce any classification result. In alternative, multiple classification results can also be provided to alert the user that the confidence in the prediction is low, or the sample can be simply treated as out-of-class.

The proposed decision rule tries to imitate a human cognitive process to perform predictions. It takes into consideration both the absolute value of the proximity scores and their relative values. This property is very useful for clinical diagnostics. In fact, in addition to the capability of identifying out-of-class samples, it can also provide important diagnostic information, such as the identification of genetic similarities with known diseases.

R1:	$\mathbf{IF}\ (\nexists i \in [1,K] \mid P\vec{V}_s[i] > Ps_{ooc})\ \mathbf{THEN}\ \mathcal{C}(\vec{s}) = y_{ooc}$
R2:	$\mathbf{IF}\ \left(\exists i \in [1,K] \mid P\vec{V}_s[i] > Ps_{max}\right)\ \mathbf{THEN}$ $\mathcal{C}(\vec{s}) = y_{\underset{i}{argmax}\left(P\vec{V}_s[i] \mid \forall i \in [1,K]\right)}$
R3:	$\mathbf{IF}\ \left(\overline{\mathbf{R1}} \wedge \overline{\mathbf{R2}} \wedge \mathbf{C}\right)\ \mathbf{THEN}\ \mathcal{C}(\tilde{\mathbf{s}}) = \mathbf{y_i}$ $\mathbf{C} = \left(\exists i,j \mid i = \underset{x}{argmax}\left(P\vec{V}_s[x] \mid \forall x \in [1,K]\right) \wedge j = \underset{x}{argmax}\left(P\vec{V}_s[x] \mid \forall x \in [1,K] \wedge x \neq i\right) \wedge \left(P\vec{V}_s[i] - P\vec{V}_s[j] > Ps_{diff}\right)\right)$
R4:	$\mathbf{IF}\ \left(\overline{R1} \wedge \overline{R2} \wedge \overline{R3}\right)\ \mathbf{THEN}\ \mathcal{C}(\vec{s}) = \mathbf{uncertain}$

Fig. 3. Decision rules for a diagnostic classifier.

The three thresholds can be defined looking at the distributions in Fig. 2:

- Ps_{max}: if the MAX plot shows a clear separation between the true and the false positive distributions, Ps_{max} can be placed in such a way to have most of the true positives immediately detected by rule R2. Ps_{max} defines the maximum probability area. Looking at Fig. 2, a good choice is $Ps_{max} \approx 0.7$. The more the threshold is placed near 1.0 the less false negatives will appear, but also the less true positives will be detected using the maximum probability rule (R2).
- Ps_{ooc}: similarly to Ps_{max}, looking at the MAX plot, Ps_{ooc} can be defined in order to correctly identify false positives using rule R1, i.e., proximity score lower than the threshold. From Fig. 2, it is clear that a good choice is $Ps_{ooc} \approx 0.3$.
- Ps_{diff}: for all samples that fall in the decision area, the DIFF graph can be used to define Ps_{diff}. A good heuristic is to consider the point where the two curves intersect. Fig. 2 shows that in general, and therefore also in the decision area, the two top classes show very close Ps for false positives, only (dotted line). In the case of true positives the distinction is much higher (between 0.3 and 0.8). $Ps_{diff} \approx 0.15$ maximizes, in this case, the number of true positives.

Whenever new data are fit into the model these parameters can be continuously tuned to improve the classification accuracy. The application of this decision rule allows us to better discriminate correct and uncertain results.

3 State-of-the-Art

Multiclass classification of gene expression profiles is a widely addressed topic literature. However, the large and complex multivariate data sets generated by microarray experiments still pose methodological and computational challenges. This section reviews a set of well-established classification methods applied in several papers to the classification of gene expression profiles and, in particular, cDNA microarray experiments.

Microarray classification can be treated as a *regression problem* where the dependent variable is categorical, and predicted from a linear combination of gene expression levels. Partial Least Squares (PLS) [17], [18] and Linear

Discriminant Analysis (LDA) [19], [20], [21] are common regression methods applied to cDNA microarray data classification.

The *k-nearest neighbors rule* (KNN) [22] is used to classify cancers based on gene expression data in several papers. KNN is an intuitive method that classifies unlabeled samples based on their similarity with examples in the training set. It finds the k-closest features in the training set and assigns the sample to the class that appears most frequently within the k-subset [18], [23], [24], [25]. The main limitations of KNN are three: storage requirements, computational cost, and extreme sensitivity to gene expression data scaling.

Decision trees [26] and *Random Forests (RF)* [27] are popular cancer classification approaches [28], [29], [30], [31]. According to Statnikov et al. [30], the capability of random forest to generate solutions from the interactivities of multiple decision trees allows to discard irrelevant genes from the classification process.

Neural Networks (NNET) are able to learn arbitrarily complex nonlinear regressions [32]. In particular, multilayer perceptrons (MLPs) are one of the most popular types of artificial neural networks that have been applied to cDNA microarray classification problems [33], [34], [35], [36].

Support vector machines (SVM) are a popular classification algorithm because of their robustness and correctness [37], [38]. Multicategory support vector machines are one of the most effective classifiers in performing accurate cancer diagnosis from gene expression data. SVMs often outperform to a remarkable degree other popular machine learning algorithms, such as KNN and NNET [39], [24], [40].

Approaches that combine multiple classifiers (*ensemble*) have also received much attention in the past decade, and are now a standard approach to improve classification performance in cDNA microarray classification problems [41], [42], [43], [44], [45], [46], [47], [48].

One of the main drawbacks of the proposed methods is the need of high-dimensionality reduction to avoid problems associated with high-dimensional sparse data sets as the one provided by microarray data. Some researchers suggested that, in microarrays classification, the choice of the dimensionality reduction method is even more important than the classification method itself [49]. Moreover, the proposed techniques are not designed to detect out-of-class samples, thus limiting their applicability in real clinical diagnostic applications.

The ability of detecting out-of-class samples, also referred to as novelty detection, outlier detection or one-class classification, is an important aspect for a machine learning system [50], [51]. Slight modifications in the data distribution might indicate, for instance, a new class or a profile modification in a class that has already been modeled. The term one-class refers to the fact that the training phase is performed on samples of a single class that represents the normal profile.

One-class classification has been used in a number of applications, especially signal processing and image analysis. Of the various approaches described in literature [50], *Parzen Window*, *KNN*, *K-Means*, and *PCA* have been applied to some extent to cDNA microarray data classification and gene expression analysis [52], [53], [54], [55]. However, combination of multiple classes into a single one, high-dimensionality feature spaces, noisy features, and quite often not enough samples make these problems hard to solve, thus reducing the efficiency of available solutions and increasing computational costs. Mitigation techniques include dimensionality reduction together with combination of different one-class classifiers as proposed in [52]. However, new dedicated solutions for the specific application domain still need to be defined.

4 Experimental Results

This section proposes an experimental validation of the GEG-based classifier. The validation represents the last step of the classifier definition process proposed in Section 2.

4.1 Experimental Design

Our experimental design involves a number of classification experiments on a large set of microarrays using both the GEG-based classifier and a collection of state-of-the-art classification methods. The GEG-based classifier has been implemented in about 2,500 lines of ANSI C code. Comparison with state-of-the-art classification methods has been performed considering two distinct problems:

1. accuracy of classifiers when working with classifiable samples, and
2. ability of dealing with out-of-class samples.

Obtained results are compared to better highlight the benefits provided by the proposed methodology.

A total of 15 pathologies is considered in this study. Samples have been downloaded from the cDNA Stanford Microarray database [56]. All genes without a valid UnigeneID have been discarded. Moreover, since old microarray technologies often used spots duplication, during the GEG generation we considered as relevant those genes relevant in at least one of their replica on the microarray.

Six sets of samples have been downloaded form a larger set of experiments aiming at performing Lymphoma Classification [57], [58], including:

- Diffuse Large B-Cell Lymphoma (DLBCL): a non-hodgkin lymphoma disease,
- B-Cell Chronic Lymphocytic Leukemia Wait & Watch (CLLww),
- B-Cell Chronic Lymphocytic Leukemia (CLL),
- Follicular Lymphoma (FL): independent lymphonode samples on LymphoChip microarrays [59],
- Hematopoietic Lymphoma (HL), and
- Normal Lymphoid subset (NL): purified normal lymphocyte subpopulations under a range of activation conditions, in normal human lymphonodes.

The remaining pathologies are:

- Acute Lymphoblastic Leukemia (ALL),
- Core Binding Factor Acute Myeloid Leukemia (CBF-AML): subgroups characterized by shorter overall survival [60],
- Breast Cancer (BC): samples of predominantly advanced primary breast tumor,

- Cutaneous B-Cell Lymphomas (CBCL): a phenotype of B-cell lymphomas of the skin,
- Healthy Blood (HB): blood samples from apparently healthy human donors. A common reference RNA (Cy3-labeled) was mixed with the Cy5-labeled experimental sample before hybridization to provide a common internal reference standard for comparison of relative gene expression levels across arrays [61],
- Solid Ovarian tumor (SOT),
- Solid Brain Tumor (SBT),
- Solid Lung Tumor (SLT), and
- Acute Myeloid Leukemia data set (AML): peripheral-blood samples or bone marrow samples of intermediate-risk AML with a normal karyotype.

Eleven out of 15 phenotypes are strictly related to blood diseases. Samples will, therefore, include very similar sets of genes and only a reduced subset of them will be able to differentiate the phenotypes. Moreover, two phenotypes (NL and HB) are related to healthy specimens. These peculiarities of the data set increase the complexity of the classification, and therefore make it a very good candidate to validate the classifier.

Table 1 shows the composition of the data set. To test the performance of our model with different cDNA technologies, we considered different types of microarray chips: 9K (9,216 spots), 18K (18,432 spots), 24K (24,168 spots), 37K (37,632 spots), and 45K (43,196 spots).

From the first nine pathologies, we extracted a training set of 213 samples and a test set of 74 classifiable samples, while the remaining nine phenotypes have been used to create a test set of 59 out-of-class samples. It is worth mentioning here that the considered training set does not include any of the samples of the test sets. According to [62], this is a given requirement to avoid overoptimistic results, and therefore to honestly evaluate the classification performances.

TABLE 1
Composition of the Experimental Data Set in Terms of Number of Test Samples (#Test), Number of Training Samples (#Training), Type of Microarray (Chip)

Classifiable samples				Out-of-class samples		
Disease	#Test	#Train	Chip	Disease	#Test	Chip
DLBCL	10	51	9k	HL	10	9k
CLLww	10	21	9k	NL	9	18k
CLL	9	12	9k	SOT	11	24k
ALL	8	19	24k	BT	12	24k
CBF-AML	7	14	45k	SLT	6	24k
BC	8	21	9k	AML	11	45k
CBCL	6	10	45k			
FL	6	18	37k			
HB	10	47	37k			
Total	**74**	**213**			**59**	

4.2 Proximity Score Analysis

The first analysis we performed on the results of the classification experiments aims at understanding whether the proximity score introduced in Section 2.2 is actually able to measure the affinity of a sample with its related class or not, and to highlight similarities among classes (diseases).

Fig. 4 graphically shows the average proximity score for the test samples of each of the 15 considered diseases (columns) against the nine considered classes (rows). The two subfigures report results obtained by using two thresholds for the identification of relevant genes (8): $\varepsilon = 0$ (1-folding) for Fig. 4a, and $\varepsilon = 1$ (2-folding) for Fig. 4b. Mismatchings are included in the average calculation. Different colors and shapes of bullets identify different ranges of the proximity score strictly related to the three decision areas used to predict the target class (see Section 2.2).

By first looking at Fig. 4a, the high score of the elements on the diagonal, together with the lowest scores of the other elements on the related columns, perfectly follow the expected outcome: each disease is correctly identified with a high proximity score in correspondence to its related class. In fact, in most of the cases, the score falls in the maximum proximity area, thus allowing to have a prediction with maximum confidence. In the three cases where the maximum proximity rule cannot be applied (ALL, BC, CBCL), all

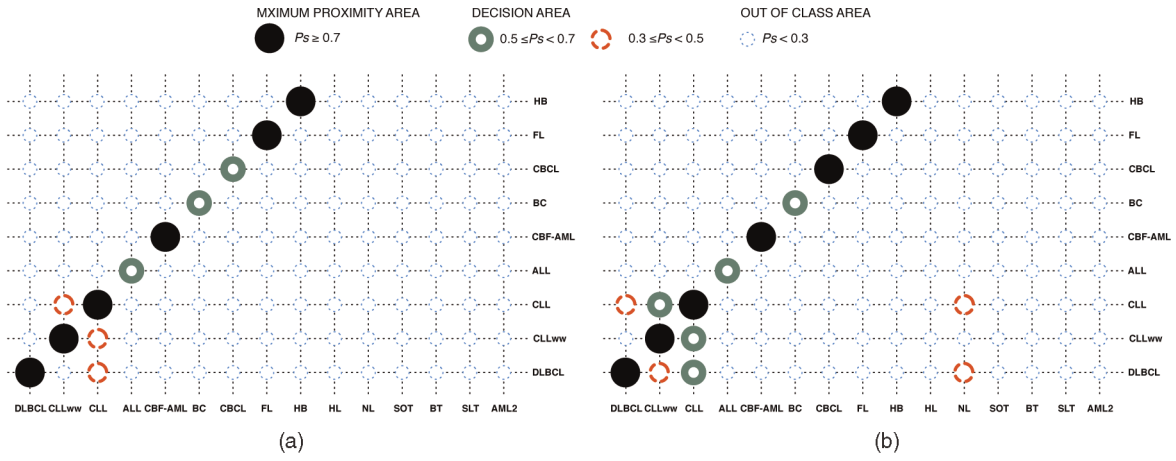

Fig. 4. Average proximity score with different gene relevance thresholds for the considered data set. Columns identify test samples grouped by diseases, while rows identify the considered classes. Bullets in the intersections of the grid represent the average proximity score of test samples of a given disease (column) for each of the available classes (rows). (a) Average proximity score with $\varepsilon = 0$ (1-folding). (b) Average proximity score with $\varepsilon = 1$ (2-folding).

TABLE 2
Average Proximity Score Computed with $\varepsilon = 0$

	DLBCL	CLLww	CLL	ALL	CBF-AML	BC	CBCL	FL	HB	HL	NL	SOT	BT	SLT	AML
DLBCL	0.776	0.279	0.459	-0.005	0.011	0.003	0.042	0.009	0.036	0.041	0.266	0.001	0.002	0.001	-0.001
CLLww	0.145	0.76	0.447	0.003	0.018	0.001	0.038	0.014	0.05	-0.006	0.159	0.001	0.003	0.001	-0.001
CLL	0.243	0.483	0.771	0.001	0.018	0.001	0.037	0.016	0.051	-0.001	0.262	0.001	0.003	0.001	-0.001
ALL	-0.001	0	0	0.656	0.14	-0.004	0.041	0.011	0.039	0.001	0	0.007	0.015	0.027	0.008
CBF-AML	0	0.001	0.001	0.078	0.901	-0.001	0.042	0.08	0.109	-0.001	0	0.012	0.025	0.021	-0.039
BC	0	0.002	0	-0.026	-0.019	0.53	0.046	-0.009	0.004	0.001	0	0.003	-0.005	-0.001	-0.002
CBCL	0.001	0.003	0.002	0.02	0.053	0.002	0.679	0	0.035	0	0.001	0.005	0.007	0.005	-0.003
FL	0.001	0	0	0.003	0.09	-0.001	-0.008	0.701	0.068	0	0	0.021	0.052	0.031	-0.085
HB	0.004	0.006	0.006	0.057	0.226	0	0.091	0.179	0.842	-0.001	0.003	0.009	0.017	0.011	-0.014

Columns identify test samples grouped by diseases, while rows identify the considered classes.

the other classes have an average proximity score in the out-of-class area, thus guaranteeing a difference higher than the chosen Ps_{diff} threshold, and therefore allowing to take a clear decision. Table 2 is another view of the same data in which it can be better appreciated the level of discrimination of the presented approach and the overall low level of noise among nonmatching sets. The only noticeable noise that appears is among NL and DLBCL, CLL and CLLww ($Ps \approx 0.23$), although the result itself is too low and does not have a sufficient discrimination w.r.t. the other scores to move the sample in one of the classes. It is also very interesting to note that the classifier is correctly able to distinguish all out-of-class samples (HL, NL SOT, BT, SLT, AML) that have been in general scored very low or even with negative scores.

In addition to this result, Fig. 4a also reveals how the proximity score is able to highlight similarities among classes. The CLLww and CLL columns show a small cluster of elements with relatively high proximity scores. This result can be reasonably interpreted as a sort of similarity among the pathologies confirmed by the fact that CLL, CLLww, and DLBCL are actually variations of a disease of blood B-cells [63]. This information is very interesting especially because it confirms how the proximity score can be used as an indicator of biological similarities of diseases. It also suggests that GEGs are able, by construction, to give more weight to genes and gene relationships that unequivocally identify a particular pathology. Obviously, this ability also depends on the quality of the training set, but this is a common problem to all supervised classification methods. Besides this property, the classifier is still able to correctly discriminate among these similar classes (see Section 4.3).

Looking at Fig. 4b, it is possible to evaluate the effect of the threshold ε used to identify relevant genes on the overall behavior of the classifier. The figure shows that the proximity score of the elements on the diagonal increases, thus increasing the confidence the classifier has in placing samples in the related classes. Moreover, the similarity among blood B-Cells is emphasized. Nevertheless, even if not producing classification errors, the noise on the NL column increases. This suggests that the used threshold was too high and probably some genes that were relevant to distinguish between NL and blood B-Cells diseases have been lost. This does not represent a major problem. In fact, the first threshold ($\varepsilon = 0$), corresponding to a situation in which genes are considered not-relevant if they have exactly the same expression in both the healthy and the diseased tissues, already produced 100 percent of correct classifications. The threshold should, therefore, be considered simply as a fine tuning of the proposed model.

4.3 GEG versus State-of-the-Art Classifiers

This section compares the performance of the GEG-based classifier with a set of state-of-the-art classification algorithms and one-class classifiers. In order to compare classification results, all experiments have been performed using log-ratios (see Section 2.1) rather than raw gene expression measures.

The analysis shows that our model has all the characteristics not only to be considered a new valid microarray classification tool, but also a very useful support for medical diagnostics.

4.3.1 Accuracy on Classifiable Samples

According to Section 3, the accuracy of the GEG-based classifier when dealing with classifiable samples has been compared with the following state-of-the-art classification algorithms applied in several papers to the classification of cDNA microarray information: k-Nearest Neighbors (KNN), Neural Networks (NNET), Linear Discriminant Analysis (LDA), Partial Least Square (PLS), Support Vector Machines (SVM), Random Forests (RF), and a set of ensemble techniques built using combinations of these approaches. Since we are dealing with classifiable samples, the keep-the-max rule has been considered for all classifiers, including the GEG-based one.

According to the literature proposed in Section 3, ensemble classifiers have been constructed on top of KNN, NNET, and RF (EKNN, ENNET, ERF). For each type of classifier we constructed as many models as the number of samples of the training set (213 in our case) using a leave-one-out (LOO) approach. The result of the ensemble classification has been then computed considering two different voting rules: majority voting (MV) and max average accuracy voting (MAAV). We finally considered a last ensemble composed of a balanced combination (ERC) of the three considered classifiers.

All algorithms have been implemented using the Classification And REgression Training (CARET) package of *R*, a free and multiplatform programming language and software environment widely used for statistical software development and data analysis [64], [65].

TABLE 3
Accuracy Estimation of Considered Classifiers, with Related Confidence Interval (CI) for 95 Percent Level of Confidence (LOC), and Specific Parameters Applied to Obtain This Result

	Avg. Acc.	CI	Specific parameters
GEG	0.964	[0.940 , 0.989]	$\epsilon = 0$
RF	0.966	[0.940 , 0.992]	mtry=20 (Number of variables randomly sampled as candidates at each split)
KNN	0.926	[0.896 , 0.956]	k = 5 (Number of nearest neighbors)
NNET	0.956	[0.903 , 1]	size=7 (Number of units in the hidden layer), decay=0.1 (Parameter for weight decay)
LDA	0.963	[0.934 , 0.992]	no pars
PLS	0.578	[0.472 , 0.685]	ncomp = 5 (Number of components one wishes to fit)
SVM	0.986	[0.923 , 1]	Radial Kernel, C=10 (cost of constraints violation) sigma=0.0214 (Inverse kernel width)
ERF (MV)	0.961	[0.938 , 0.984]	Ensemble of 213 RF classifiers with majority voting decision rule (MV)
EKNN (MV)	0.892	[0.861 , 0.923]	Ensemble of 213 KNN classifiers with majority voting decision rule (MV)
ENNET (MV)	0.964	[0.933 , 0.995]	Ensemble of 213 NNET classifiers with majority voting decision rule (MV)
ERC (MV)	0.946	[0.901 , 0.991]	Ensemble of 213 classifiers: 71 RF, 71 KNN, 71NNET with majority voting decision rule (MV)
ERF (MAAV)	0.949	[0.928 , 0.970]	Ensemble of 213 RF classifiers with Max Average Accuracy Voting (MAAV)
EKNN (MAAV)	0.842	[0.808 , 0.876]	Ensemble of 213 KNN classifiers with Max Average Accuracy Voting (MAAV)
ENNET (MAAV)	0.598	[0.547 , 0.649]	Ensemble of 213 NNET classifiers with Max Average Accuracy Voting (MAAV)
ERC (MAAV)	0.698	[0.657 , 0.739]	Ensemble of 213 classifiers: 71 RF, 71 KNN, 71NNET with Max Average Accuracy Voting (MAAV)

The performance of the prediction model of each classifier has been tuned and optimized by performing leave-group-out-cross-validation (LGOCV). For each classifier, 30 folds of the training set have been generated with 98 percent of samples used to train the model, while the remaining ones used as test set. The size of the grid used to search the tuning parameters space for each classifier (e.g., k for KNN) has been set to 5. This represents a good compromise in terms of computational time of the training phase and optimization results.

Table 3 summarizes the average accuracy estimated from the cross-validation experiments for each considered classifier, together with the specific parameters applied to reach this results. For each estimated accuracy the corresponding confidence interval (CI) computed with 95 percent level of confidence (LOC) is also provided.

The accuracy obtained on the considered test set is reported in Table 4 for all considered classifiers. According to Fig. 5, that shows in a single diagram both the estimated accuracy with related CI and the test-set results, the GEG-based classifier performs as the best state-of-the-art algorithms.

TABLE 4
Classification Results for Classifiable Samples Using the Keep-the-Max Rule for All Classifiers

Classifier	Matches	Mismatches
GEG	73/74 (98.65%)	1/74 (1.35%)
RF	73/74 (98.65%)	1/74 (1.35%)
KNN	69/74 (93.24%)	5/74 (6.76%)
NNET	70/74 (94.59%)	4/74 (5.41%)
LDA	73/74 (98.65%)	1/74 (1.35%)
PLS	42/74 (56.76%)	32/74 (43.24%)
SVM	73/74 (98.65%)	1/74 (1.35%)
ERF (MV)	73/74 (98.65%)	1/74 (1.35%)
EKNN (MV)	68/74 (91.89%)	6/74 (8.11%)
ENNET (MV)	73/74 (98.65%)	1/74 (1.35%)
ERC (MV)	73/74 (98.65%)	1/74 (1.35%)
ERF (MAAV)	71/74 (95.95%)	3/74 (4.05%)
EKNN (MAAV)	64/74 (86.49%)	10/74 (13.51%)
ENNET (MAAV)	46/74 (62.16%)	28/74 (37.84%)
ERC (MAAV)	53/74 (71.62%)	21/74 (28.38%)

4.3.2 Ability of Identifying Out-of-Class Samples

When considering both classifiable and out-of-class samples, the GEG-based classifier outperforms state-of-the-art methods for out-of-class samples detection. In this case, the GEG-based classifier applies the decision rule presented in Fig. 3.

Results are provided in terms of true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), sensitivity, specificity, and f-score. Sensitivity, specificity, and f-score are statistical measures of the performance of a binary classification test. The *sensitivity* measures true positives, i.e., the proportion of classifiable samples that are correctly identified as such (e.g., the percentage of sick people who are identified as having a disease), the *specificity* measures true negatives, i.e., the proportion of correctly

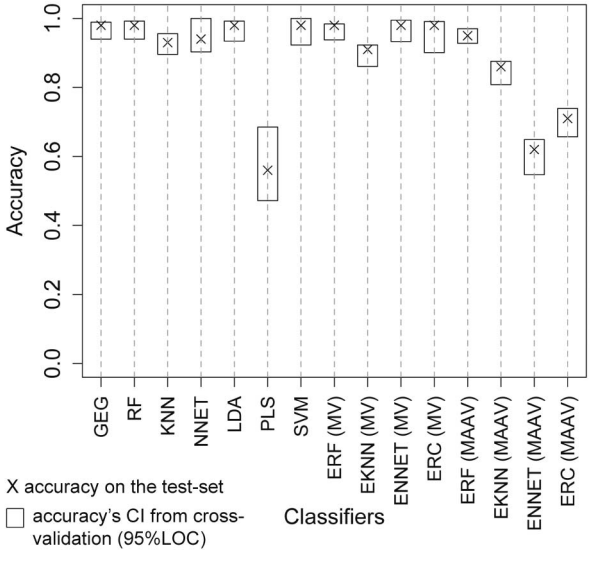

Fig. 5. Comparison of accuracy and confidence intervals ($\mathrm{LOC} = 95$ percent) for the considered classifiers.

TABLE 5
Confusion Matrix for the GEG-Based Classifier with the Considered Test Set ($Ps_{ooc} = 0.3$, $Ps_{max} = 0.7$, and $Ps_{diff} = 0.15$)

	Classifiable	Out-of-class
Classifiable	70/74 with 1M (94.59% TP)	0/59 (0% FP)
Out-of-class	4/74 with 1U (5.41% FN)	59/59 with 5 U (100% TN)

identified out-of-class samples (e.g., the percentage of healthy people who are identified as not having any disease), and the *f-score* is a measure of the test's accuracy that considers both precision (number of correct results divided by the number of returned results) and recall (number of correct results divided by the number of results that should have been returned).

Table 5 proposes a confusion matrix that summarizes the performance of the GEG-based classifier when dealing with both classifiable and out-of-class samples of the considered test set. The most positive result is that 100 percent of out-of-class samples (TN column) are correctly identified. This aspect is a critical issue in biological classification and, in particular, in clinical diagnostics. U indicates that 5 out of 59 samples have been placed in this category but the decision rule produced an *uncertain* result to warn the user that the confidence in these predictions is not maximum. The cost of the application of the decision rule is a very reduced amount of FN. Four out of 74 samples have been erroneously classified as out-of-class (with one of these classified as uncertain). This, of course, derives from the impossibility of perfectly separating the different decision areas with the considered thresholds. Nevertheless, results are quite promising, with the sensitivity equal to 0.95, specificity equal to 1, and f-score equal to 0.96.

The prediction model of the GEG-based classifier when working with out-of-class samples has been cross-validated by performing the same set of experiments proposed in Section 2.2.1 to identify the decision rule thresholds. Table 6 shows the average sensitivity and specificity together with the related confidence intervals with $\mathrm{LOC} = 95$ percent. The first row reports the results of the complete cross-validation experiments. These results show a good estimate of the sensitivity that confirms the experimental results on the test set, while they significantly downestimate the sensitivity. The reason of this bias is in the type of performed cross-validation experiments. The considered training set (Table 1) contains two classes corresponding to two variants of the same disease (CLL and CLLww). Whenever during the cross-validation one of these two classes is removed from the library, detecting its samples as out-of-class becomes a very hard problem for the classifier, increasing the number of FP.

TABLE 6
Estimation of Sensitivity and Specificity of the GEG-Based Classifier by Performing Cross-Validation Experiments

Cross. Val.	Sens.	Sens. CI	Spec.	Spec. CI
Complete	0.945	[0.931 , 0.960]	0.718	[0.547 , 0.867]
No Cll, Cllww	0.947	[0.932 , 0.961]	0.906	[0.824 , 0.989]

Confidence intervals are provided with $LOC = 95$ percent.

TABLE 7
Estimation of Sensitivity and Specificity of One-Class Classifiers by Performing Cross-Validation Experiments

Cross. Val.	Sens.	Sens. CI	Spec.	Spec. CI
Parzen*	0.254	[0.230 , 0.280]	0.744	[0.640, 0.848]
KNN	0.938	[0.920 , 0.956]	0.03	[0 , 0.066]
KMeans	0.829	[0.80 , 0.86]	0.155	[0.084 , 0.227]
PCA	0.254	[0.23 , 0.28]	0.744	[0.638 , 0.850]
E1 (MV)	0.826	[0.796 , 0.856]	0.159	[0.087 , 0.231]
E2 (MV)	0.826	[0.796 , 0.856]	0.159	[0.087 , 0.231]
E1 (MSV)	0.254	[0.227 , 0.281]	0.744	[0.638 , 0.850]
E2 (MSV)	0.306	[0.28 , 0.34]	0.0692	[0 , 0.180]

Confidence intervals are provided with $LOC = 95$ percent.

This is confirmed by the second row of Table 6. In this case, the sensitivity and specificity are estimated without considering CLL and CLLww as out-of-class samples. This downestimation of the specificity of the classifier when performing cross-validation may also bias the selection of the thresholds proposed in Section 2.2.1. Different cross-validation approaches are under investigation to overcome these problems.

The performance of the GEG-based classifier in detecting out-of-class samples has been compared with a set of four state-of-the-art one-class classification methods, namely, Parzen, KNN, KMeans, and PCA (see Section 3), all implemented using Matlab's DD_tools [66]. The target class has been constructed using the nine classes of classifiable samples with 5 percent rejection rate. According to [52], we also considered two ensembles of one-class classifiers: the first one E1 composed of Parzen, KNN, and KMeans, while the second E2 composed of KNN, KMeans, and PCA. Two different voting rules have been applied for the ensembles: majority voting (MV), and max score voting (MSV). Table 7 proposes the result of the same type of cross-validation experiments applied to the GEG-based classifier (No CLL, CLLww), while Table 8 summarizes the obtained classification results on the test set for all classifiers, including the GEG-based one.

Fig. 6 graphically compares in a set of Receiver Operating Characteristic (ROC) curves, the results for all considered classifiers. The ROC curve for the GEG-based classifier has been computed by varying the three thresholds applied in the proposed decision rules, while the one for the one-class classifier is computed by changing the considered rejection boundary. The black dots represent the performance of the classifier in terms of true positive

TABLE 8
One-Class Classification Results

Classifier	TP	TN	FN	FP	Sens	Spec	F-score
GEG	**70**	**59**	**4**	**0**	**0.95**	**1**	**0.96**
Parzen	**44**	**39**	**30**	**20**	**0.59**	**0.66**	**0.64**
KNN	74	0	0	59	1	0	0.71
KMeans	72	0	2	59	0.97	0	0.70
PCA	44	30	30	29	0.59	0.51	0.60
E1 (MV)	72	0	2	59	0.97	0	0.70
E2 (MV)	72	0	2	59	0.97	0	0.70
E1 (MSV)	44	39	30	20	0.59	0.66	0.64
E2 (MSV)	57	7	17	52	0.77	0.11	0.62

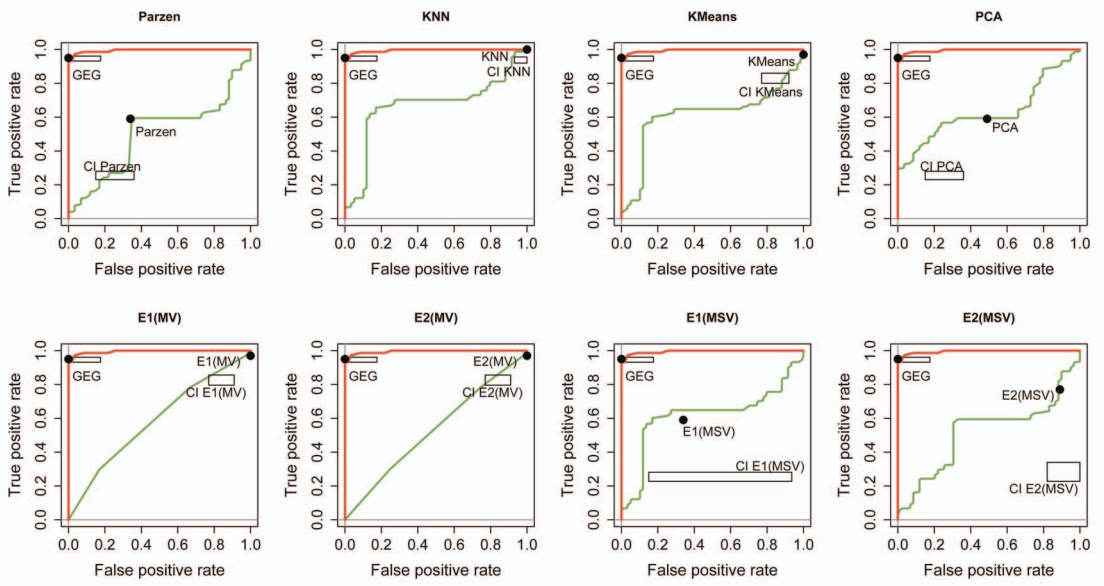

Fig. 6. GEG-based classifier versus one-class classifier. Black dots represent performances on the considered test set.

TABLE 9
Classification Results After Filtering Out-of-Class Samples with the Parzen One-Class Classifier

	RF	KNN	NNET	LDA	PLS	SVM	E_KNN MV	E_NNET MV	E_RF MV	E_KNN MAAV	E_NNET MAAV	E_RF MAAV	E_RC MV	E_RC MAAV
Mismatches	1	5	2	1	19	1	5	1	1	8	21	3	1	15
	2.27%	11.36%	4.55%	2.27%	43.18%	2.27%	11.36%	2.27%	2.27%	18.18%	47.73%	6.82%	2.27%	34.09%

rate (i.e., sensitivity) and false positive rate (i.e., 1-specificity) on the considered test set, while the squares represent the confidence intervals of the true/false positive rates obtained from the cross-validation.

It is evident how the proposed approach is the one that better allows to drastically reduce the number of FP and FN conditions that would make a diagnostic tool unusable in real medical setups. Moreover, it provides results on the test set that are more coherent with the one obtained by cross-validating the model. This is an important characteristic to early estimate the performance of a given classifier based on the available training data.

Among the different one-class classifiers, Parzen seems to provide the best compromise in terms of sensitivity, specificity, f-score, and computational complexity. Therefore, to deal with both classifiable and out-of-class samples, we combined this one-class classifier with the state-of-the-art multiclass classification methods proposed in Table 4. The one-class classifier is used to filter out-of-class samples, while classifiable samples are then submitted to the considered classifiers. Table 9 reports how the number (and percentage) of mismatches changes when samples are first filtered by the one-class classifier. The reduction of the number of mismatches should not be considered here as an improvement of the classifiers' accuracy. In fact, it is correlated to the fact that the one-class classifier erroneously rejects some of those classifiable samples that would generate mismatchings.

4.3.3 Computational Time

One important aspect to take into account when comparing the performance of different classifiers is the overall computation time. Several steps are required to complete the classification of a test set: data preprocessing (e.g., Near-Zero-Variance (NZV) used to identify and eliminate near-zero-variance genes in the data set and PCA used for dimensionality reduction), training, and classification. Table 10 summarizes the execution time required by each step for the GEG-based classifier, and for the combination of state-of-the-art classifiers with the Parzen one-class classifier.

Table 10 illustrates the only real criticality of the proposed approach. In fact, while the training phase for the GEG-based classifier is negligible, thus allowing an easy update of the classification model, the classification time is an order of magnitude higher than the one of the other classifiers. Two are the main motivations for this difference. First, the GEG-based classifier works without performing any dimensionality reduction (see Section 4.3.4); while this has positive effects on the classification performances, it strongly increases the computational costs. Second, the available implementation is a preliminary prototype and several optimizations can be introduced in the code to reduce the computation time and increase its efficiency.

TABLE 10
Comparison of Execution Time for the Different Classifiers

	GEG	PLS + One-Class	Others + One-Class	Ensemble + One-Class
pre-processing	-	36′19″	~38′09″	38′09″
Training	00′04″	05′33″	~03′04″	312′40″
Classification	103′00″	05′19″	~02′28″	5′36
TOTAL	**103′04″**	**48′11″**	**43′41″**	**356′25′**

TABLE 11
Comparison of Data Dimensionality Reduction between the GEG-Based and the State-of-the-Art Classifiers

	GEG	PLS	Others	One-Class
Start Genes	62,300*	62,300*	62,300*	62,300*
After reduction	**59,878 genes**	**5PC**	**77 PC**	**232 PC**

* Because of the gene overlapping among microchips

4.3.4 Data Reduction

In microarray analysis, genes are the information. Dimensionality reduction is usually applied for making the problem treatable by the available computational setup, but there is no guarantee that the discarded data may not become useful or interesting. The GEG approach does not need heavy dimensionality reduction. This allows to keep most of the information in the GEGs, making it easier to reuse the same data model for further analysis of the gene expression data. An example of this is the development of clustering and feature extraction algorithms, able to extract from the GEG structure the pathological genetic markers, again by analyzing the topological structure and the weights of the GEGs. Clearly, limiting the dimensionality reduction impacts on the execution time, but never to a point to make the algorithm unusable (even without any dimensionality reduction as can be seen in Section 4.3.3).

The number of variables, either log-ratios of gene expression levels or principal components (PC), considered by each tested classifier is summarized in Table 11. It clearly appears how the number of variables maintained in the GEG after the training is far higher than in all other methods.

5 Conclusions and Future Works

In this paper, we have presented a new classification algorithm designed to be used in real clinical diagnostic applications. The classifier is based on a new graph-based data model used to organize large amounts of gene expression data coming from microarray experiments. This new data model is very flexible, and it makes the implementation of classification, clustering, and feature extraction algorithms easier and less "abstracted" from the biological problem. The classifier is not only able to correctly classify samples in the corresponding classes, but it is also able to correctly detect out-of-class samples, thus drastically reducing the false positive rate. Experiments on a set of cDNA microarrays provided very good results. In order to reduce the computational time, one of the main limitations of the proposed tool, and to make it freely available to other researchers for additional testing and validation, we are currently working on implementing the proposed classification into the CARET package of *R*.

References

[1] G. Gibson, "Microarray Analysis," *PLoS Biology,* vol. 1, no. 1, pp. 28-29, Oct. 2003.

[2] P. Larranaga, B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, J.A. Lozano, R. Armananzas, A. Santafe, G. ad Perez, and V. Robles, "Machine Learning in Bioinformatics," *Briefings in Bioinformatics,* vol. 7, no. 1, pp. 86-112, Feb. 2006.

[3] E.R. Dougherty, "The Fundamental Role of Pattern Recognition for Gene-Expression/Microarray Data in Bioinformatics," *Pattern Recognition,* vol. 38, no. 12, pp. 2226-2228, Dec. 2005.

[4] A. Benso, S. Di Carlo, G. Politano, and L. Sterpone, "A Graph-Based Representation of Gene Expression Profiles in DNA Microarrays," *Proc. IEEE Symp. Computational Intelligence in Bioinformatics and Computational Biology (CIBCB),* pp. 75-82, Sept. 2008.

[5] A. Benso, S. Di Carlo, G. Politano, and L. Sterpone, "Differential Gene Expression Graphs: A Data Structure for Classification in DNA Microarrays," *Proc. Eighth IEEE Int'l Conf. BioInformatics and BioEng. (BIBE),* pp. 1-6, Oct. 2008.

[6] D. Jiang, C. Tang, and A. Zhang, "Cluster Analysis for Gene Expression Data: A Survey," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 11, pp. 1370-1386, Nov. 2004.

[7] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. Altman, "Missing Value Estimation Methods for DNA Microarrays," *Bioinformatics,* vol. 17, no. 6, pp. 520-525, June 2004.

[8] A. Hill, E. Brown, M. Whitley, G. Tucker-Kellog, C. Hunter, and D.K. Slonim, "Evaluation of Normalization Procedures for Oligonucleotide Array Data Based on Spiked cRNA Controls," *Genome Biology,* vol. 2, no. 12, Nov. 2001.

[9] J. Schuchhardt, D. Beule, A. Malik, E. Wolski, H. Eickhoff, H. Lehrach, and H. Herzel, "Normalization Strategies for cDNA Microarrays," *Nucleic Acids Research,* vol. 28, no. 10, p. E47, May 2000.

[10] Unigene, http://www.ncbi.nlm.nih.gov/sites/entrez?db=unigene, 2010.

[11] J. Stuart, E. Segal, D. Koller, and S. Kom, "A Gene-Coexpression Network for Global Discovery of Conserved Genetic Modules," *Science,* vol. 302, no. 5643, pp. 249-255, Oct. 2003.

[12] D.B. Allison, X. Cui, G.P. Page, and M. Sabripour, "Microarray Data Analysis: From Disarray to Consolidation to Consensus," *Nature Rev.: Genetics,* vol. 7, no. 1, pp. 55-65, May 2006.

[13] M.K. Kerr, M. Martin, and G.A. Churchill, "Analysis of Variance for Gene Expression Microarray Data," *J. Computational Biology,* vol. 7, no. 6, pp. 819-837, Dec. 2000.

[14] C. Cheadle, M.P. Vawter, W.J. Freed, and K.G. Becker, "Analysis of Microarray Data Using Z Score Transformation," *J Molecular Diagnostics,* vol. 5, no. 2, pp. 73-81, 2003.

[15] D.M. Witten and R. Tibshirani, "A Comparison of Fold-Change and the t-Statistic for Microarray Data Analysis," http://www-stat.stanford.edu/tibs/ftp/FCTComparison.pdf, 2009.

[16] E. Parzen, "On Estimation of a Probability Density Function and Mode," *The Ann. Math. Statistics,* vol. 33, no. 3, pp. 1065-1076, 1962.

[17] D. Nguyen and D. Rocke, "Tumor Classification by Partial Least Squares Using Microarray Gene Expression Data," *Bioinformatics,* vol. 18, no. 1, pp. 39-50, Jan. 2002.

[18] J.W. Lee, J.B. Lee, M. Park, and S.H. Song, "An Extensive Comparison of Recent Classification Tools Applied to Microarray Data," *Computational Statistics and Data Analysis,* vol. 48, no. 4, pp. 869-885, 2005.

[19] L. Sheng, R. Pique-Regi, S. Asgharzadeh, and A. Ortega, "Microarray Classification Using Block Diagonal Linear Discriminant Analysis with Embedded Feature Selection," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing,* pp. 1757-1760, 2009.

[20] Y. Guo, T. Hastie, and R. Tibshirani, "Regularized Linear Discriminant Analysis and Its Application in Microarrays," *Biostatistics,* vol. 8, no. 1, pp. 86-100, 2007.

[21] P. Xu, G.N. Brock, and R.S. Parrish, "Modified Linear Discriminant Analysis Approaches for Classification of High-Dimensional Microarray Data," *Computational Statistics and Data Analysis,* vol. 53, no. 5, pp. 1674-1687, 2009.

[22] *Nearest Neighbor (NN) Norms: Nn Pattern Classification,* B.V. Dasarathy, ed., IEEE CS, 1991.

[23] L. Li, C.R. Weinberg, T.A. Darden, and L.G. Pedersen, "Gene Selection for Sample Classification Based on Gene Expression Data: Study of Sensitivity to Choice of Parameters of the GA/KNN Method," *Bioinformatics,* vol. 17, no. 12, pp. 1131-1142, 2001.

[24] A. Statnikov, C.F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy, "A Comprehensive Evaluation of Multicategory Classification Methods for Microarray Gene Expression Cancer Diagnosis," *Bioinformatics,* vol. 21, no. 5, pp. 631-643, 2005.

[25] J. Liang and S. Kachalo, "Computational Analysis of Microarray Gene Expression Profiles: Clustering, Classification, and Beyond," *Chemometrics and Intelligent Laboratory Systems,* vol. 62, no. 2, pp. 199-216, 2002.

[26] J. Breiman, L. ad Friedman, C.J. Stone, and R. Olshen, *Classification and Regression Trees*. Taylor and Francis, Inc, 1984.
[27] L. Breiman, "Random Forests," *Machine Learning*, vol. 1, no. 45, pp. 5-32, 2001.
[28] H. Zhang, C.-Y. Yu, and B. Singer, "Cell and Tumor Classification Using Gene Expression Data: Construction of Forests," *Proc. Nat'l Academy of Science USA*, vol. 100, no. 7, pp. 4168-4172, Apr. 2003.
[29] H. Zhang, C.-Y. Yu, B. Singer, and M. Xiong, "Recursive Partitioning for Tumor Classification with Gene Expression Microarray Data," *Proc. Nat'l Academy of Science USA*, vol. 98, no. 12, pp. 6730-6735, Jun. 2001.
[30] A. Statnikov, L. Wang, and C. Aliferis, "A Comprehensive Comparison of Random Forests and Support Vector Machines for Microarray-Based Cancer Classification," *BMC Bioinformatics*, vol. 9, no. 1, p. 319, 2008.
[31] R. Diaz-Uriarte and S. Alvarez de Andres, "Gene Selection and Classification of Microarray Data Using Random Forest," *BMC Bioinformatics*, vol. 7, no. 1, p. 3, 2006.
[32] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*, second ed. Wiley-Interscience, 2000.
[33] F. Azuaje, "A Computational Neural Approach to Support the Discovery of Gene Function and Classes of Cancer," *IEEE Trans. Biomedical Eng.*, vol. 48, no. 3, pp. 332-339, Mar. 2001.
[34] C.-J. Huang and W.-C. Liao, "Application of Probabilistic Neural Networks to the Class Prediction of Leukemia and Embryonal Tumor of Central Nervous System," *Neural Processing Letters*, vol. 19, no. 3, pp. 211-226, 2004.
[35] 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*, vol. 7, no. 6, pp. 673-679, June 2001.
[36] G. Bloom, I.V. Yang, D. Boulware, K.Y. Kwong, D. Coppola, S. Eschrich, J. Quackenbush, and T.J. Yeatman, "Multi-Platform, Multi-Site, Microarray-Based Human Tumor Classification," *Am. J. Pathology*, vol. 164, no. 1, pp. 9-16, 2004.
[37] V.N. Vapnik, "An Overview of Statistical Learning Theory," *IEEE Trans. Neural Networks*, vol. 10, no. 5, pp. 988-999, Sept. 1999.
[38] G. Natsoulis, L. El Ghaoui, G.R. Lanckriet, A.M. Tolley, F. Leroy, S. Dunlea, B.P. Eynon, C.I. Pearson, S. Tugendreich, and K. Jarnagin, "Classification of a Large Microarray Data Set: Algorithm Comparison and Analysis of Drug Signatures," *Genome Research*, vol. 15, no. 5, pp. 724-736, 2005.
[39] K. Crammer and Y. Singer, "On the Algorithmic Implementation of Multiclass Kernel-Based Vector Machines," *J. Machine Learning Research*, vol. 2, pp. 265-292, 2002.
[40] 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, no. 10, pp. 906-914, 2000.
[41] A.-L. Boulesteix, C. Porzelius, and M. Daumer, "Microarray-Based Classification and Clinical Predictors: On Combined Classifiers and Additional Predictive Value," *Bioinformatics*, vol. 24, no. 15, pp. 1698-1706, 2008.
[42] L. Frey, M. Edgerton, D. Fisher, and S. Levy, "Ensemble Stump Classifiers and Gene Expression Signatures in Lung Cancer," *Studies in Health Technology and Informatics*, vol. 129, no. Pt 2, pp. 1255-1259, 2007.
[43] J.-H. Hong and S.-B. Cho, "The Classification of Cancer Based on DNA Microarray Data that Uses Diverse Ensemble Genetic Programming," *Artificial Intelligence in Medicine*, vol. 36, no. 1, pp. 43-58, Jan. 2006.
[44] B. Liu, Q. Cui, T. Jiang, and S. Ma, "A Combinational Feature Selection and Ensemble Neural Network Method for Classification of Gene Expression Data," *BMC Bioinformatics*, vol. 5, p. 136, Sept. 2004.
[45] Y. Peng, "A Novel Ensemble Machine Learning for Robust Microarray Data Classification," *Computers in Biology and Medicine*, vol. 36, no. 6, pp. 553-573, June 2006.
[46] A. Statnikov, C.F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy, "A Comprehensive Evaluation of Multicategory Classification Methods for Microarray Gene Expression Cancer Diagnosis," *Bioinformatics*, vol. 21, no. 5, pp. 631-643, Mar. 2005.
[47] K.-J. Kim and S.-B. Cho, "Ensemble Classifiers Based on Correlation Analysis for DNA Microarray Classification," *Neurocomputing*, vol. 70, nos. 1-3, pp. 187-199, 2006.
[48] S.B. Cho and H.-H. Won, "Cancer Classification Using Ensemble of Neural Networks with Multiple Significant Gene Subsets," *Applied Intelligence*, vol. 26, no. 3, pp. 243-250, 2007.
[49] S. Deegalla and H. Boström, "Classification of Microarrays with kNN: Comparison of Dimensionality Reduction Methods," *Proc. Eighth Int'l Conf. Intelligent Data Eng. and Automated Learning (IDEAL)*, pp. 800-809, 2007.
[50] M. Markou and S. Singh, "Novelty Detection: A Review—Part 1: Statistical Approaches," *Signal Processing*, vol. 83, pp. 2481-2497, 2003.
[51] V. Hodge and J. Austin, "A Survey of Outlier Detection Methodologies," *Artificial Intelligence Rev.*, vol. 22, no. 2, pp. 85-126, Oct. 2004.
[52] E. Spinosa and A. de Carvalho, "Combining One-Class Classifiers for Robust Novelty Detection in Gene Expression Data," *Advances in Bioinformatics and Computational Biology*, pp. 54-64, Springer-Verlag, 2005.
[53] X. Yun and R.G. Brereton, "Diagnostic Pattern Recognition on Gene-Expression Profile Data by Using One-Class Classification," *J. Chemical Information and Modeling*, vol. 45, no. 5, pp. 1392-1401, 2005.
[54] P. Juszczak, D.M. Tax, E.P. Kalska, and R.P. Duin, "Minimum Spanning Tree Based One-Class Classifier," *Neurocomputing*, vol. 72, nos. 7-9, pp. 1859-1869, 2009.
[55] V. Gesù, G. Bosco, and L. Pinello, "A One Class Classifier for Signal Identification: A Biological Case Study," *Proc. 12th Int'l Conf. Knowledge-Based Intelligent Information and Eng. Systems, Part III (KES '08)*, pp. 747-754, 2008.
[56] cDNA Stanford's Microarray Database, http://genome-www.stanford.edu/, 2010.
[57] A.A. Alizadeh, M.B. Eisen, R.E. Davis, C. Ma, I.S. Lossos, A. Rosenwald, J.C. Boldrick, H. Sabet, T. Tran, X. Yu, J.I. Powell, L. Yang, G.E. Marti, T. Moore, J.J. Hudson, L. Lu, D.B. Lewis, R. Tibshirani, G. Sherlock, W.C. Chan, T.C. Greiner, D.D. Weisenburger, J.O. Armitage, R. Warnke, R. Levy, W. Wilson, M.R. Grever, J.C. Byrd, D. Botstein, P.O. Brown, and L.M. Staudt, "Distinct Types of Diffuse Large B-Cell Lymphoma Identified by Gene Expression Profiling," *Nature*, vol. 403, no. 6769, pp. 503-511, Feb. 2000.
[58] C. Palmer, M. Diehn, A. Alizadeh, and P.O. Browncorresponding, "Cell-Type Specific Gene Expression Profiles of Leukocytes in Human Peripheral Blood," *BMC Genomics*, vol. 7, no. 115, 2006.
[59] S.P. Bohen, O.G. Troyanskaya, O. Alter, R. Warnke, D. Botstein, P.O. Brown, and R. Levy, "Variation in Gene Expression Patterns in Follicular Lymphoma and the Response to Rituximab," *Proc. Nat'l Academy of Science USA*, vol. 100, no. 4, pp. 1926-1930, Feb. 2003.
[60] L. Bullinger et al., "Gene-Expression Profiling Identifies Distinct Subclasses of Core Binding Factor Acute Myeloid Leukemia," *Blood*, vol. 110, no. 4, pp. 1291-1300, 2007.
[61] A.R. Whitney, M. Diehn, S.J. Popper, A.A. Alizadeh, J.C. Boldrick, D.A. Relman, and P.O. Brown, "Individuality and Variation in Gene Expression Patterns in Human Blood," *Proc. Nat'l Academy of Science USA*, vol. 100, no. 4, pp. 1896-1901, Feb. 2003.
[62] C. Ambroise and G.J. McLachlan, "Selection Bias in Gene Extraction on the Basis of Microarray Gene-Expression Data," *Proc. Nat'l Academy of Science USA*, vol. 99, pp. 6562-6566, 2002.
[63] A. Rosenwald, A.A. Alizadeh, G. Widhopf, R. Simon, R.E. Davis, X. Yu, L. Yang, O.K. Pickeral, L.Z. Rassenti, J. Powell, D. Botstein, J.C. Byrd, M.R. Grever, B.D. Cheson, N. Chiorazzi, W.H. Wilson, T.J. Kipps, P.O. Brown, and L.M. Staudt, "Relation of Gene Expression Phenotype to Immunoglobulin Mutation Genotype in B Cell Chronic Lymphocytic Leukemia," *J. Experimental Medicine*, vol. 194, no. 11, pp. 1639-1648, 2001.
[64] R Development Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2010. [Online]. Available: http://www.R-project.org.
[65] K. Kuhn, "Building Predictive Models in R Using the Caret Package," *J. Statistical Software*, vol. 28, no. 5, pp. 1-26, Aug. 2008.
[66] D.M.J. Tax, "Ddtools, the Data Description Toolbox for Matlab," http://www-ict.ewi.tudelft.nl/davidt/dd_tools.html, 2010.

Alfredo Benso received the MS degree in computer engineering and the PhD degree in information technologies, both from Politecnico di Torino, Italy, where he is working as a tenured associate professor of computer engineering. His research interests include pattern recognition techniques for bioinformatics. He is also actively involved in the Computer Society, where he has been a leading volunteer for several projects. He is a Computer Society Golden Core Member, and a senior member of the IEEE.

Stefano Di Carlo received the MS degree in computer engineering and the PhD degree in information technologies from Politecnico di Torino, Italy, where he is working as an assistant professor in the Department of Control and Computer Engineering since 2008. His research interests include pattern recognition techniques for bioinformatics and chemical sensors. He is a Golden Core member of the IEEE Computer Society and a member of the IEEE.

Gianfranco Politano received the MS degree in computer engineering from Politecnico di Torino, Italy, where he is working toward the PhD degree in the Department of Control and Computer Engineering since 2008. His main research interests are microarray data analysis and pattern recognition techniques for bioinformatics. He is a student member of the IEEE and the IEEE Computer Society.