BIOINFORMATICS ORIGINAL PAPER Vol. 22 no. 17 2006, pages 2114–2121
doi:10.1093/bioinformatics/btl346

Gene expression

Detecting potential labeling errors in microarrays by data perturbation

Andrea Malossini[1,*], Enrico Blanzieri[1] and Raymond T. Ng[2]

[1]Department of Information and Communication Technology, University of Trento, 38050 Povo, Italy and
[2]Department of Computer Science, University of British Columbia, Vancouver, BC V6T1Z4, Canada

Received on November 28, 2005; revised on May 28, 2006; accepted on June 22, 2006
Advance Access publication July 4, 2006
Associate Editor: Satoru Miyano

ABSTRACT

Motivation: Classification is widely used in medical applications. However, the quality of the classifier depends critically on the accurate labeling of the training data. But for many medical applications, labeling a sample or grading a biopsy can be subjective. Existing studies confirm this phenomenon and show that even a very small number of mislabeled samples could deeply degrade the performance of the obtained classifier, particularly when the sample size is small. The problem we address in this paper is to develop a method for automatically detecting samples that are possibly mislabeled.

Results: We propose two algorithms, a classification-stability algorithm and a leave-one-out-error-sensitivity algorithm for detecting possibly mislabeled samples. For both algorithms, the key structure is the computation of the leave-one-out perturbation matrix. The classification-stability algorithm is based on measuring the stability of the label of a sample with respect to label changes of other samples and the version of this algorithm based on the support vector machine appears to be quite accurate for three real datasets. The suspect list produced by the version is of high quality. Furthermore, when human intervention is not available, the correction heuristic appears to be beneficial.

Contact: malossin@dit.unitn.it

1 INTRODUCTION

With the advances of high-throughput devices for measuring gene and protein expression, numerous medical or health research groups have amassed large amounts of genomic data. The hope is that such data can be used to infer regulatory pathways in cells, identify novel targets for drug design and improve the diagnosis, prognosis and treatment planning for those suffering from the disease. However, many groups find out that the analysis of such data is not as straightforward as initially expected. There are in general three main challenges for analyzing such data.

- High dimensionality p: high-throughput devices for measuring gene or protein expression collect a large number of measurements, or variables, per patient. For example, the number of probe sets in the Affymetrix U133 Plus 2.0 Array is $p > 54000$.
- Small sample size n: it is not uncommon for medical applications involving human samples to have small sample sizes, by which we mean $n < 100$ [e.g. West *et al*. (2001); Golub *et al*. (1999); Vapnik *et al*. (2002); Alon *et al*. (1999); Alizadeh *et al*. (2000); Ramaswamy *et al*. (2003)]. One reason is the availability of patients and samples [e.g. early stage lung cancer analysis Chan *et al*. (2004)]. Another reason is the potential high cost of preparing and maintaining the samples in the 'wet' laboratory (e.g. micro-dissection and RNA extraction) as well as the cost for the preparation of microarray experiments.
- Potential labeling errors: Classification is widely used in medical applications. However, the quality of the classifier depends critically on the accurate labeling of the training data[1]. But for many medical applications, labeling a sample or grading a biopsy can be subjective. For example, West *et al*. (2001) analyzed 49 breast tumor samples and identified 9 samples as possibly having the wrong labels.

The problem we address in this paper is to develop a method for automatically detecting samples that are possibly mislabeled. As shown in Malossini *et al*. (2005), even a very small number of mislabeled samples could deeply degrade the performance of the obtained classifier. Furthermore, the smaller the sample size, the greater the impact of a mislabeled sample.

The problem of detecting uncertainty in data labeling has been approached in many different directions. Brodley and Friedl (1999) used a set of different classifiers that serve as noise filters for the training data. Then the final classifier is trained on the training set where instances identified as uncertain are removed. Muhlenbach *et al*. (2004) proposed a filtering algorithm where the suspect samples were removed or relabeled before the learning stage. An example is considered suspect when in its neighbourhood, defined by a geometrical graph, the proportion of examples of the same class is not significantly greater than in the database itself. Sanchez *et al*. (2003) proposed different methods, based on the nearest neighbour classifiers, in order to improve the quality of the training data and to reduce the overlapping among regions of different classes. Venkataraman *et al*. (2004) used a single optimal classifier, the SVM with a linear kernel on multiple representations of the data. Each representation is built by choosing different subspaces of the whole feature space, then a leave-one-out (LOO) cross-validation is used to identify mislabeled data. However, in all these approaches,

*To whom correspondence should be addressed.

[1] In this paper, we only focus on binary classification; the idea could be generalized to a multi-class situation.

the datasets used were characterized as having $n > p$. It is unlikely that any of these approaches work effectively for the different, and arguably tougher, situation when $p \gg n$.

One possible approach is to apply dimensionality reduction algorithms (such as principal component analysis), or feature selection algorithms to first reduce the dimensionality of the data. But it is well-known that dimensionality reduction algorithms do not perform well for unclean data (outliers or mislabeled samples can biased the resulting list of features); see for example the study by De la Torre and Black (2001). Similarly, standard feature selection algorithms assume that all samples are correctly labeled, and can be significantly affected by the presence of mislabeled samples.

Some mislabeled samples may be detected as outliers. (In general, a mislabeled sample need not be outlying, and an outlier is not necessarily mislabeled.) Barnett and Lewis (1994) give a survey of outlier detection techniques in conventional statistics. Outlier detection for high-dimensional data has received a lot of attention in recent years. For instance, Knorr and Ng (1998) considered distance-based outlier detection, where the notion of outlier is strongly dependent on the underlying distance metric. Aggarwal and Yu (2001) studied the problem of outlier detection for high-dimensional data using projections into subspaces. However, this approach is clearly not scalable for large p. Furthermore, to use a distance-based approach, one has to define a distance function between samples. In a high-dimensional space, it is often difficult to do so. Thus, an outlier detection method that requires an explicit definition of distance may not be effective for the situations we are dealing with here.

To overcome this compounding situation of high dimensionality with a small number of samples, our approach does not rely on distance computation. Instead, our approach is based on examining classifiers in combination with a LOO procedure and a flipping strategy. More specifically, we make the following contributions.

We introduce the Leave-One-Out Perturbed Classification matrix (LOOPC matrix). To compute the entry LOOPC[i, j], we first flip the label $y_i \in \{-1, +1\}$ of the sample x_i. Then leaving sample x_j out, a classifier is built with the flipped sample x_i, and is then used to predict the label of x_j. We choose to use a state-of-the-art kernel algorithm, the support vector machine (SVM), in its simplest form, where the kernel used is linear. The column LOOP[$*$, j] contains the prediction for x_j based on different perturbed datasets, whereas the row LOOPC[i, $*$] are the predictions of different samples based on the same perturbed datasets with x_i flipped.

We then propose two algorithms for analyzing the LOOPC matrix. The first algorithm, called the Classification-stability algorithm (CL-stability), focuses on analyzing each column to produce a list of suspect samples. The main idea behind this analysis is to assess the stability (of the classification) of a sample with respect to a small perturbation (just one flip) of the other samples of the dataset. Good samples should be consistently classified even when a small perturbation is introduced. The second algorithm, called the Leave-One-Out-Error-sensitivity algorithm (LOOE-sensitivity), focuses on analyzing each row to produce a list of suspect samples. The main idea behind this row analysis is that if the sample is mislabeled, then flipping it should improve the prediction power of the resulting classifier.

Algorithm 1 Compute the LOO Perturbed Classification matrix L

Require: $S := \{(x_k, y_k)\colon 1 \le k \le n\}$
1: **for** $i := 1$ to n **do**
2: $S_i :=$ Flip(S, i)
3: **for** $j := 1$ to n **do**
4: $L[i, j] := \mathcal{C}_{S_i}^{-j}(x_j)$
5: **end for**
6: **end for**

We provide an empirical evaluation of the two algorithms using real and synthetic datasets. Under various circumstances, we evaluate the precision and recall performance of the algorithms whenever 'ground truth' is available. By the term 'ground truth' we mean here a list of suspect samples which have been validated using biological knowledge or tests. In short, the classification-stability algorithm performs well with real datasets.

While the two algorithms are designed for detecting possibly mislabeled samples, the remaining issue is how to recover once those samples have been identified. For situations when no human intervention is possible, we propose a simple heuristic to rebuild a new classifier. Our preliminary experimental results using real datasets indicate that this simple heuristic can improve prediction accuracy.

2 METHODS

2.1 Algorithm for creating the LOOPC matrix

The basic ingredient of our algorithms is the construction of the LOOPC matrix. The main idea behind such matrix is to flip the label of a single sample in the training set, to construct a classifier, and finally to apply the resulting perturbed classifier to the left-out sample. Hereafter we assume that there are n samples of the form (x_i, y_i) where x_i is the vector of features and y_i is the label of the sample. We denote the LOOPC matrix as L, which is an $n \times n$ matrix. Algorithm 1 describes the construction of the LOOP matrix, where the Flip function changes the label of the sample (we are here considering only binary classification problems), and $\mathcal{C}_{S_i}^{-j}(x_j)$ denotes the classification outcome of a classifier trained on $S_i \backslash (x_j, y_j)$ and applied to x_j. In this way each row of the resulting matrix represents the effect of the perturbation on the sample (x_i, y_i) to the whole dataset. Each column represents the behaviour of the sample (x_j, y_j) with respect to the flipping of other samples of the dataset. Note that the diagonal elements represent the unperturbed LOO classification of the samples. At the end of the algorithm, the LOOPC matrix is as follows:

$$L = \begin{pmatrix} \mathcal{C}_{S_1}^{-1}(x_1) & \mathcal{C}_{S_1}^{-2}(x_2) & \dots & \mathcal{C}_{S_1}^{-n}(x_n) \\ \mathcal{C}_{S_1}^{-1}(x_1) & \mathcal{C}_{S_2}^{-2}(x_2) & \dots & \mathcal{C}_{S_2}^{-n}(x_n) \\ & & \vdots & \\ \mathcal{C}_{S_n}^{-1}(x_1) & \mathcal{C}_{S_n}^{-2}(x_2) & \dots & \mathcal{C}_{S_n}^{-n}(x_n) \end{pmatrix}.$$

It is easy to see that the algorithm requires n^2 training phases and testing phases. However, because we are dealing with 'small' datasets, this procedure is still computationally feasible. It should be obvious that Algorithm 1 is easily parallelizable.

As a very simple example, consider five patients subject to a diagnosis of a rare disease. Suppose that patients 1 and 3 are diagnosed as having the disease D and that patients 2, 4 and 5 are diagnosed as healthy H. To construct the first row of L, the state of the first patient is flipped from D to H. Then the LOO procedure is applied to the other patients to predict the label of the left-out patients with the resulting classifiers. For example, the first row shown below indicates that, when perturbing the diagnosis

Algorithm 2 CL-stability

Require: $L[i, j]$ matrix
Require: $S := \{(x_k, y_k): \quad 1 \leq k \leq n\}$
for $j = 1$ to n **do**
  $n_+ := |\{i : L[i, j] = +1, \quad 1 \leq i \leq n\}|$
  $n_- := |\{i : L[i, j] = -1, \quad 1 \leq i \leq n\}|$
  if $[(n_+ \geq \alpha_n) \wedge (y_j = -1)] \vee [(n_- \geq \alpha_n) \wedge (y_j = +1)]$ **then**
    Sample (x_j, y_j) is a suspect
  else
    Sample (x_j, y_j) is not a suspect
  end if
end for

of the first patient, all the labels remain correct, except for patient 2. The remaining rows are similarly obtained.

$$L = \begin{pmatrix} D & D & D & H & H \\ D & H & D & H & H \\ D & D & D & D & H \\ D & D & D & H & H \\ D & D & D & H & H \end{pmatrix}.$$

2.2 The CL-stability algorithm for analyzing L

Given the LOOPC matrix L, the goal of applying the CL-stability algorithm is to assess the stability of a sample with respect to a small perturbation (just one flip) of the other samples of the dataset. A good or stable sample should be consistently classified, with respect to its original label, and not be easily affected by the correctness of 1 flipped sample elsewhere in the dataset. The CL-stability algorithm identifies a list of samples failing this requirement.

To be specific, for a given sample x_j, the quantities n_+ and n_- count the number of times that x_j is classified as '+1' and '−1' respectively. We use '+1' and '−1' to represent the two possible states of classification. Sample x_j is considered a statistically significant suspect of mislabeling if the number of mis-classifications exceeds a certain threshold, which is determined as follows. We consider a set of n completely independent random classifiers; their outcomes are '+1' or '−1'. Thus, the distribution can be modeled as a binomial distribution with $p = 0.5$. We say that a sample is a suspect if the mis-classification frequency deviates from the mean by >3 SD [this is a condition frequently used in outlier detection in conventional statistics, see Barnett and Lewis (1994)]. The exact formula is defined as

$$\alpha_n = \frac{n}{2} + \frac{3}{2}\sqrt{n}.$$

2.2.1 Aggregated predictors and CL-stability algorithm The results of the CL-stability algorithm is an aggregated prediction [see e.g. Breiman (1996) for the theory of bagging predictors]. Bagging predictors is a method for aggregating different predictors where the learning is performed on bootstrap replicates of the training set. The resulting predictor outcome is a majority vote across the bootstrapped predictors. Aggregating can transform good predictors into nearly optimal one and bad predictors into worse one. Moreover, bagging unstable classifier usually improves them. In this scenario, the CL-stability algorithm can be viewed as a bootstrapping procedure where the bootstrap training set is substituted with a perturbed training set. Also the voting scheme is different, because in order to account for the flipping of the labels a threshold is calculated (α_n) which permits to discriminate the random classifier.

2.3 The LOOE-sensitivity algorithm for analyzing L

The LOOPC matrix can also be analyzed exploiting the information contained in each row of the matrix. For a given sample x_i, the i-th row

Algorithm 3 LOOE-sensitivity

Require: $L[i, j]$ matrix
Require: $S := \{(x_k, y_k) : 1 \leq k \leq n\}$
for $i := 1$ to n **do**
  neutralP := countP := countN := neutralN := 0
  for $j := 1$ to n **do**
    if $j \neq i$ **then**
      if $y_j = L[i, j] \wedge y_j = L[j, j]$ **then**
        neutralP++
      end if
      if $y_j = L[i, j] \wedge y_j \neq L[j, j]$ **then**
        countP++
      end if
      if $y_j \neq L[i, j] \wedge y_j = L[j, j]$ **then**
        countN++
      end if
      if $y_j \neq L[i, j] \wedge y_j \neq L[j, j]$ **then**
        neutralN++
      end if
    end if
  end for
  Output x_i as a suspect if (countP-countN) $\geq \beta_n$
end for

of L gives the classification of x_i (for each $j := 1, \ldots, n$) if the label of x_j flipped. The LOOE-sensitivity algorithm produces a list of samples with suspect labeling by measuring whether flipping the label significantly improves prediction accuracy. More specifically, it considers four cases:

(1) x_j is correctly predicted without flipping x_i (i.e. $y_j = L[j, j]$), and the prediction of x_j remains correct even after x_i is flipped (i.e. $y_j = L[i, j]$);
(2) x_j is incorrectly predicted without flipping x_i (i.e. $y_j \neq L[j, j]$, but the prediction becomes correct with a flip of x_i (i.e. $y_j = L[i, j]$);
(3) this is the reverse of the above situation case 2 when flipping x_i makes an initial correct prediction of x_j incorrect and
(4) the prediction of x_j remains incorrect whether x_i is flipped or not.

By flipping x_i, samples in the second case give concrete 'positive' evidence for flipping x_i. However, samples in the third case correspond to 'negative' evidence. Samples in the remaining two cases are 'neutral' because prediction accuracy is unchanged whether x_i is flipped or not. The last step of the LOOE-sensitivity algorithm is to return all the samples where the difference between positive and negative evidence exceeds a threshold $\beta_n \cdot \beta_n$ is a parameter of the algorithm chosen by the user.

2.4 Datasets and parameter settings

We tested our algorithms on three different well-known real datasets: (1) A breast cancer dataset from West *et al*. (2001), which consists of 49 tumor samples classified as positive to estrogen receptor ER+ or negative ER−. The expression levels of 7129 genes are given for each sample. (2) A colon tissue dataset from Alon *et al*. (1999) which consists of 40 tumor samples and 22 normal samples. We used the same 2000 genes selected in the study. (3) A leukemia dataset from Golub *et al*. (1999), which consists of 25 acute myeloid leukemia and 47 acute lymphoblastic leukemia. The expression levels of 7129 genes are given for each sample.

The samples identified by West *et al*. (2001) and Alon *et al*. (1999) are used as 'ground truth', whereas for the leukemia dataset no ground truth is available. Moreover, we compare the results obtained in Furey *et al*. (2000), Kadota *et al*. (2003) and Li *et al*. (2001). Furey *et al*. (2000) used a SVM on

the full datasets (tuning a diagonal factor to achieve the best performance on leave-one-out cross-validation test) and on a specified top-ranked features. Samples which have been consistently misclassified in all tests are identified as suspects. Kadota *et al*. (2003) faced the problem of detecting outliers using a technique based on Akaike's Information Criterion. Li *et al*. (2001) used a genetic algorithm in order to select subsets of genes that can potentially discriminate between tumor and normal tissue samples and then each sample is classified according to the class membership of its k nearest neighbors.

According to Alon *et al*. (1999) some of the samples of the colon cancer dataset have been identified as outliers (using a hierarchical clustering), namely T2, T30, T33, T36, T37, N8, N12, N34. Moreover, these samples presented an anomalous muscle-index which confirms the uncertainty of these samples. The muscle-index is the average over the intensity of 17 ESTs in the array that were homologous to smooth muscle genes; normal tissue should have a high muscle-index whereas cancer tissue should have a low muscle-index [see Alon *et al*. (1999)]. The sample N36 has been considered anomalous because of its low muscle index. Other works [Furey *et al*., 2000; Kadota *et al*., 2003; Li *et al*. 2001] identify some suspect samples in the colon cancer and leukemia datasets using solely statistical methods or machine learning algorithm without any biological or clinical confirmation. For the breast cancer dataset, as stated in West *et al*. (2001), the assay of ER status by immunohistochemistry is far from perfect and can produce erroneous results. In fact five samples of the dataset, namely the number 14, 31, 33, 45, 46, the immunohistochemistry diagnosis conflicted with a successive immunoblot assay. Samples 16, 40, 43 labeled as ER- presented instead elevated levels of several known estrogen-regulated genes. Sample 11 instead was uncertain.

Real benchmark datasets with biologically ground truth are not easy to obtain. Thus, there are a lot of situations that may arise in practice that cannot be tested. Synthetic datasets were designed to simulate these situations. For instance, the number of dimensions, p, and the number of distinguishing features may vary from one situation to another. Furthermore, the two distributions may overlap to varying degrees. Using synthetic datasets allows us to more exhaustively evaluate how the algorithms would behave under different conditions. Moreover, in the real datasets, we may not truly know the ground truth. In synthetic datasets, the truths are known. This makes examination of the algorithms more concrete. Evaluation against these synthetic datasets complements the evaluation using real datasets.

In Table 1 we show the name of the synthetic datasets and the corresponding settings of the parameters. The flipped samples are selected randomly and they may come from one or both classes. In all the experiments, we used a SVM with a linear kernel. We chose not to perform model selection because possibly mislabeled samples could badly influence the construction of the LOOPC matrix. In fact the model selection procedure could give very low weight to the slack variables of the SVM allowing for very large errors (hence, the effect of flipping would be cancelled), on the other side, giving too much weight to the slack variables could excessively influence the construction of the hyperplane (hence, the effect of the flipping would be too strong). Since we have no control over the two possibilities we decided to set $C = 1$ in all the simulations in order to compare the results using the same value for C. As a comparison (on real datasets only), we also used a local classifier, a nearest neighbor classifier, with $k = 3$, in which the classification is based on majority voting. We decided to use the k-NN classifier because, based on the article Dudoit *et al*. (2002), it performs well with respect to others simpler linear classifier in microarray classification tasks. For the implementation of the Furey's algorithm we decided to follow the indication in Furey *et al*. (2000) and run the algorithm on the full datasets and on a reduced dataset of 3500 features (note that the choice of the number of features to select is arbitrary) with different values for the diagonal factor $0, 0.1, 0.5, 1, 1.5$. Finally, in the LOOE-sensitivity algorithm β_n is set to 1.

Table 1. Synthetic dataset used

Dataset	p	$f_{\pm}$	μ_{+}	σ_{+}	μ	σ
Synthetic2000	2000	20	3	1	−3	3
Synthetic1000	1000	20	3	1	−3	3
Synthetic200	200	20	3	1	−3	3
Synthetic100	100	20	3	1	−3	3
Synthetic2000sep	2000	20	30	1	−30	3
Synthetic2000ovr	2000	20	3	1	0	3

For a two-class classification problem, a synthetic dataset was generated by sampling, for each feature, from a uni-variate gaussian distribution with the same mean μ and standard deviation σ. The two classes differed only for a limited number of features $f_{\pm}$ drawn from a gaussian distribution with a different mean $\mu_{\pm}$ and a narrower standard deviation σ_{+}. The number of samples was fixed to $n = 30$ and the datasets generated were balanced. The first four combinations are identical except that the dimensionality p varies from 2000 to 100. The last two combinations capture the situations when the two classes are either strongly separated or overlapped.

All the algorithms have been implemented using the language for statistical computing R (http://www.R-project.org).

3 RESULTS AND DISCUSSION

3.1 Synthetic datasets

We first used the synthetic datasets described above to evaluate the CL-stability and LOOE-sensitivity algorithms. Table 2 shows precision, recall and number of identified suspects of the CL-stability and LOOE-sensitivity algorithms, averaged across 100 trials. In each trial we randomly flip 0% (for the first row) and 10% of the samples. For example, for the second row of the table, we generated 100 different instances of Synthetic2000; the mean values are obtained by averaging across the 100 instances. In each trial, we recorded the number of suspect samples identified, the precision and the recall. It is possible to compute precision and recall values because we know precisely which and how many samples were mislabeled (the fourth column gives this number). The first row of Table 2 is for dataset instances with 2000 features, 20 of which were distinguishing, but none of the samples had been flipped. The CL-stability algorithm identified 1.2 suspect samples on average. The next four rows are for dataset instances all with 20 distinguishing features and 3 out of 30 samples flipped. The only difference among the four rows is the variation in p, the number of features, from 2000 to 100. As p decreases and the number of distinguishing features remain fixed at 20, the two classes become less similar relatively speaking, making the identification of mislabeling easier. This is reflected in the improvement of both the precision and recall values produced by the CL-stability algorithm.

The next three rows of Table 2 are for dataset instances all with 2000 features and 3 flipped samples. The only difference is the variation in the number of distinguishing features, which changes from 100 to 5. (The row corresponding to the case with 20 distinguishing features is duplicated from a previous row for easier comparison.) As expected, both the precision and recall values of the CL-stability algorithm decrease as the number of distinguishing features is reduced.

Finally, the last two rows of the first part of Table 2 represent dataset instances all with 2000 features, 20 distinguishing features

Table 2. Mean precision, recall and number of suspects (100 random trials) when using the CL-stability algorithm and the LOOE-sensitivity algorithm, some samples are artificially flipped

Dataset	n	Number of distinguishing features	Number of mislabeled samples	Cl-stability Prec	Rec	Number of suspects	LOOE-sensitivity Prec	Rec	Number of suspects
Synthetic2000	30	20	0	—	—	1.2	—	—	2.2
Synthetic2000	30	20	3	0.37	0.56	5.4	0.20	0.60	10.2
Synthetic1000	30	20	3	0.60	0.74	4.1	0.25	0.59	7.4
Synthetic200	30	20	3	0.89	0.95	3.3	0.31	0.55	5.5
Synthetic100	30	20	3	0.88	0.93	3.4	0.30	0.57	6.1
Synthetic2000	30	100	3	1.00	0.99	3.0	0.04	0.15	2.0
Synthetic2000	30	20	3	0.37	0.56	5.4	0.20	0.60	10.2
Synthetic2000	30	5	3	0.11	0.32	9.1	0.11	0.43	12.9
Synthetic2000sep	30	20	3	0.58	0.70	4.4	0.27	0.63	9.8
Synthetic2000ovr	30	20	3	0.17	0.37	7.9	0.14	0.47	11.2
Breast cancer†	40	n.d.	4	0.78 (0.16)	0.92 (0.12)	5.0	0.28 (0.23)	0.47 (0.33)	7.7
Colon cancer†	51	n.d.	5	0.62 (0.14)	0.91 (0.14)	7.7	0.30 (0.25)	0.51 (0.32)	10.2
Leukemia†	70	n.d.	7	0.60 (0.10)	0.91 (0.12)	10.8	0.24 (0.16)	0.50 (0.26)	18.5

The first part of the table describes the performance of the two proposed algorithms on different synthetic datasets. The second part regards real datasets that have been cleansed (we removed all the samples identified as mislabeled or suspects in at least one study in the literature, see Section 3.2). The symbol † denotes a cleansed real dataset. In parenthesis we report the standard deviation.

and 3 flipped samples. The only difference is how widely separated the two classes were, as described earlier. As expected, both the precision and recall values of the CL-stability algorithm improve as the separation becomes wider.

In general, for the CL-stability algorithm, the recall value is higher than the corresponding precision value. We believe that from a mislabeling identification point of view, it would be better this way than if the recall value is lower. We view the identification exercise as a way to cleanse the data. For example, the list of suspects may be given to an expert to perform additional assessment of the samples. Thus, for many applications, it is better to err on the conservative side in having false positives than having false negatives.

Next let us turn to the LOOE-sensitivity algorithm. The figures reported in Table 2 for the algorithm are for the situation when the positive evidence strictly outweighs the negative evidence (i.e. $\beta_n = 1$). In general, if we raise the threshold β_n, precision would improve but at the expense of recall. Conversely, if we lower β_n, the reverse relationship applies.

It is obvious that in almost every situation, the CL-stability algorithm dominates the LOOE-sensitivity algorithm, particularly with respect to recall values. The idea of the CL-stability algorithm is to measure how the labeling of the current sample is affected by the labeling of other samples. In contrast, the LOOE-sensitivity algorithm measures how the labeling of the current sample affects the labeling of other samples. From the results shown thus far, it is apparent that the former strategy is far more effective in identifying mislabeled samples. In the following, we present results on the real datasets. Again, the LOOE-sensitivity algorithm gives results dominated by the CL-stability algorithm. Thus, we only show the results of the CL-stability algorithm below.

3.2 Real datasets

The second part of Table 2 shows the performance of CL-stability and LOOE-sensitivity algorithms on the cleansed datasets (we remove the samples which have been identified as suspects in at least one work in the literature). For all the three real datasets, the performance of the CL-stability dominates those of the LOOE-sensitivity, thus strengthening the choice of using the CL-stability for further analysis.

In Table 3 we report, for three real datasets, the samples identified as suspects using different algorithms. For each dataset a different 'ground truth' is used: Alon *et al*. for the colon cancer, West *et al*. (2001) for the breast cancer and for the leukemia dataset we used the samples identified by the majority of other works (because no biological validation is available). Since the CL-stability algorithm is not restricted to a particular classification strategy we evaluated its performance using as core algorithm a SVM and a 3-nearest neighbor. As further comparisons, we used a simple LOO detecting procedure (the misclassified samples are suspects), and for the breast cancer dataset we applied the algorithm used in Furey *et al*. (2000). Note that even among human analysts, there may not be universal consensus as to which samples are mislabeled or not. For the colon cancer, the CL-stability algorithm using SVM correctly identifies six out of nine suspects, producing only two false positives. The simple LOO identifies also the sample N12 but at the cost of four false positives. Furey instead identifies six samples with no false positives. For the breast cancer the situation is similar but here the CL-stability using SVM correctly identifies five out of nine samples with no false positives, whereas the other algorithms produce one or two false positives. For the leukemia dataset all the algorithms agree on identifying the sample 66 as suspects but Furey generates another suspect and the CL-stability algorithm using 3-NN generates five more suspects. In general, the CL-stability algorithm using the SVM gives satisfactory results. For all three datasets, the suspect list appears to be of high quality, rivaling the suspect list generated by human experts using more accurate biological techniques (e.g. immunoblotting). The version of the CL-stability algorithm using 3-NN does not perform as well. From Table 3 results that the CL-stability

Table 3. Lists of suspects obtained on the original real datasets

Dataset	Source	Suspects identified w.r.t. the gold standard									Extra samples identified
Colon cancer	Alon *et al*. (1999)	T2	T30	T33	T36	T37	N8	N12	N34	N36	
	CL-stability (SVM)	✓	✓	✓	✓				✓	✓	N2, N28
	CL-stability (3-NN)		✓		✓		✓		✓	✓	N2, N7, N27, N39
	Simple LOO	✓	✓	✓	✓			✓	✓	✓	T8, N2, N28, N29
	Furey *et al*. (2000)		✓	✓	✓		✓		✓	✓	
	Kadota *et al*. (2003)	✓				✓	✓		✓	✓	T6, N2
	Li *et al*. (2001)		✓	✓	✓				✓	✓	
Breast cancer	West *et al*. (2001)	11	14	16	31	33	40	43	45	46	
	CL-stability (SVM)		✓	✓	✓		✓	✓			
	CL-stability (3-NN)		✓	✓	✓		✓				19, 36
	Simple LOO		✓	✓	✓		✓	✓			19, 47
	Furey Algorithm		✓	✓	✓		✓	✓			47
Leukemia	Majority	66									
	CL-stability (SVM)	✓									
	CL-stability (3-NN)	✓									2,29,35,47,64
	Simple LOO	✓									
	Furey *et al*. (2000)	✓									54
	Li *et al*. (2001)	✓									

The first line for each dataset represents the gold standard; whereas for the colon and breast cancer there is a biological validation of the suspect highlighted, for the leukemia dataset such validation does not exist (hence the gold standard is given by the samples highlighted by the majority of the algorithm considered).

algorithm and the Furey's algorithms are comparable in terms of detection power of suspects. However it has to be noticed that the latter algorithm implies an arbitrary choice for the number of features to be selected and for the range of the diagonal factor. Moreover it has been shown in Malossini *et al*. (2005) that mislabeled samples can considerably affect the resulting list of selected genes, hence the results obtained using those lists. On the contrary, our CL-stability algorithm use the dataset 'as it is' and the parameter α_n is automatically chosen.

Table 4 shows the number of suspects identified using both the SVM and the 3-NN versions on the cleansed datasets. Again, the version using SVM performs very satisfactorily—with only one false positive for the breast cancer dataset, and with no identified suspect for the other two datasets. In contrast, the version using 3-NN does not perform as well, particularly with six false positives for the leukemia dataset.

One way to differentiate between the SVM version and the 3-NN version is that the former can be viewed as 'global', whereas the latter can be viewed as 'local'. In the local case, a single flip may have significant influence within the neighbourhood. Thus, it may tend to be overly aggressive, leading to poor precision. In contrast, a more global strategy, such as SVM, uses information contained in all the samples. As such, it may be more conservative in identifying suspects. Since we generally expect that mislabeled samples are rare events, the more conservative approach appears to be more effective. From now on, we only show the results using the SVM version of the CL-stability algorithm.

Finally, it might be interesting to plot the receiver operating characteristics (ROC) curves of the CL-stability as a function of the α_n parameter, see Figure 1, using as ground truth the elements identified by Alone *et al*. (1999) and West *et al*. (2001). The filled circles denote the value of α_n chosen using the argumentation explained in Section 2.2.

Table 4. Number of suspects generated in the analysis of cleansed datasets using the CL-stability algorithm

Cleansed dataset	Number of suspects	
	SVM	3-NN
Breast cancer (40)	1	2
Colon cancer (51)	0	1
Leukemia (70)	0	6

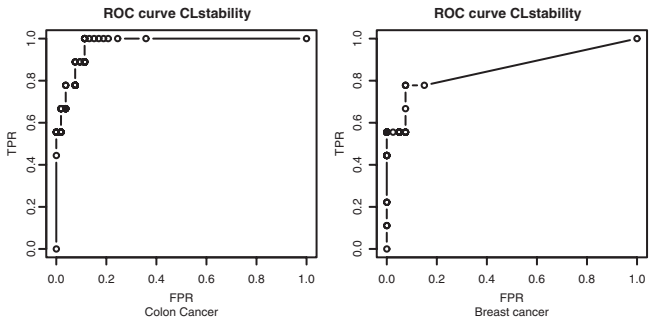

Fig. 1. ROC curves for the CL-stability algorithm. For the colon cancer the list provided by Alone *et al*. (1999) is used as ground truth; for the breast cancer the list provided by West *et al*. (2001) is used as ground truth. The parameter α_n is varying from $n+1$ to 0 in step of 1; the filled circle represents the threshold chosen using the given formula for α_n

3.3 An automatic correction heuristic

So far, our evaluation focuses on identification of suspect samples. An immediate question to ask is what we expect the user to do with the suspect list. We intend the CL-stability algorithm

Table 5. Accuracy and comparison between the original classifier and the different classifiers obtained when using the three methods for handling suspect samples

Methods	Colon cancer			Breast cancer			Leukemia		
	Accuracy	CE	BH	Accuracy	CE	BH	Accuracy	CE	BH
Original dataset (orig)	0.81	0.00	0	0.86	0.00	0	0.99	0.00	0
Cleansed dataset using the literature (cleansed)	1.00	0.10	57	0.98	0.08	40	0.99	0.01	19
Corrected dataset using literature (lit-corr)	0.90	0.18	80	0.86	0.18	73	0.93	0.03	31
Corrected dataset using the heuristic (heur-corr)	0.89	0.19	72	0.96	0.10	61	0.99	0.01	27

LOO accuracy of the classifier obtained.

to serve as an 'alert' filter for notifying the user about the quality of the samples. Thus, the expected course of action is for the user to investigate further the suspects. However, for some applications, such human intervention may not be available. In that case, we propose a simple correction heuristic–that we reverse the label of each sample in the suspect list. In the following, we evaluate the effectiveness of this heuristic.

To compare between the different classifiers obtained, we used two metrics. The first one is the classification error of the classifier with respect to the original labels and it is defined as

$$\mathrm{CE} = \frac{|\{i : \mathcal{C}(x_i) \neq y_i, \quad 1 \leq i \leq n\}|}{n},$$

where $\mathcal{C}(x_i)$ gives the label of applying classifier $\mathcal{C}$ (trained on the corresponding dataset) to the sample x_i. The second metric is independent of labels where we compute the angles BH between the hyperplane of the original classifier and the hyperplanes obtained for the other classifiers. Table 5 compares four classifiers: the SVM classifier based on the original dataset, the SVM classifier based on the cleansed dataset (remove all the samples identified as suspect in the literature), the SVM classifier based on the corrected dataset using the literature (flip back all the samples identified as suspect in the literature) and the SVM classifier based on the corrected dataset where, as suggested in the proposed heuristic, the label of a sample in the suspect list is flipped. There are three columns in the table, giving the LOO accuracy (average across all the samples), the disagreement figure CE and the BH angle. Considering the disagreement figures and the angles, it is clear that the corrected classifiers are different from the corresponding original classifiers for all three datasets. More importantly, the accuracy of the corrected classifiers is better than the original classifiers for the colon cancer and the breast cancer datasets. (The leukemia dataset does not discriminate among the three classifiers.) Note that the accuracies presented are computed on the same dataset used for the identification of the suspects, hence all the accuracies may contain a small bias. Therefore those accuracies should not be regarded as an unbiased estimate of the generalization error. In the absence of human intervention, the corrected classifiers often behave almost as well as the classifiers based on the cleansed datasets. This shows that the quality of the suspect lists is high, and the automatic correction heuristic is reasonable. Furthermore, for situations where the sample size is already small, the correction heuristic may help to preserve as many samples as possible, thereby preserving the statistical power of the analysis.

To evaluate the impact of the correction heuristic, we use feature selection as a way to compare between the original dataset and the corrected dataset. Feature selection is a meaningful task because for many medical applications, the researchers are interested in the actual genes that discriminate between the two classes, not simply to predict the class label. We use the recursive feature elimination using SVM (Vapnik *et al*., 2002) and for each iteration we eliminate 10% of the genes. Figure 2 shows the comparison of the selected features for the colon and the breast datasets. The x-axis gives the top-k genes/features selected, where k varies from 20 to 200. The y-axis gives the number of selected features in common (i.e. the size of the intersection of the top-k lists). The plots A and C show the size of the intersection between the original dataset and the literature-corrected dataset and the intersection between the corrected dataset and the literature-corrected dataset. Similarly, the plots B and D show the two curves with respect to the cleansed dataset. It is clear that list produced when we flip back or remove the suspects can be very different from the list obtained using the original datasets. Moreover, flipping back has a stronger effect on the resulting list than removing them, with respect to the list obtained using the original dataset. This observation is confirmed for the breast dataset. In fact, the gap between the corrected dataset and the original dataset is even wider for the breast data.

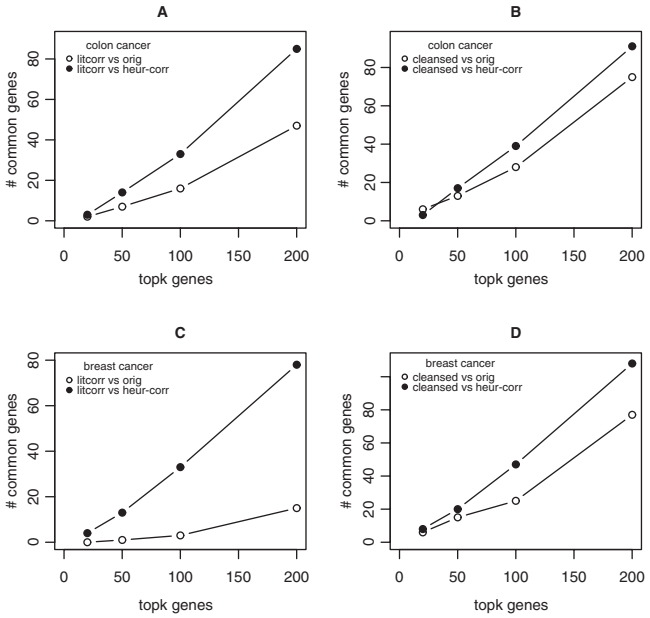

Fig. 2. Common genes when using the RFE on the different datasets listed in Table 5.

4 CONCLUSION

In this paper, we propose two algorithms for detecting possibly mislabeled samples. The CL-stability algorithm is based on measuring the stability of the label of a sample with respect to label changes of other samples. The version of the CL-stability algorithm based on SVM appears to be quite accurate for three real datasets. The LOOE-sensitivity algorithm which relies on assessing the effect of a single perturbation on the LOO error does not perform well. The good performance achieved in the CL-stability may be owing to the fact that in this algorithm different perturbations are analyzed for a single sample, whereas in the LOOE-sensitivity the effect of only a single perturbation is analyzed. The suspect list produced by the SVM version is of high quality. Furthermore, when human intervention is not available, the correction heuristic appears to be beneficial. In future work, we believe that the correction heuristic can be improved further. A more sophisticated heuristic could selectively flip the labels of some, but not all, of the samples in the suspect list.

ACKNOWLEDGEMENTS

The authors would like to thank the associate editor and the anonymous referees for their comments that helped to improve the paper.

Conflict of interest: none declared.

REFERENCES

Aggarwal,C.C. and Yu,P.S. (2001) Outlier detection for high dimensional data. In *Proceedings of ACM SIGMOD 2001,* Santa Barbara, CA, pp. 37–46.

Alizadeh,A. *et al*. (2000) Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, **403**, 503–511.

Alon,U. *et al*. (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotides array. *Proc. Natl Acad. Sci. USA*, **96**, 6745–6750.

Barnett,V. and Lewis,T. (1994) *Outliers in Statistical Data*. John Wiley and Sons, New York.

Breiman,L. (1996) Bagging predictors. *Mach. Learn*., **26**, 123–140.

Brodley,C.E. and Friedl,M.A. (1999) Identifying mislabeled training data. *J. Artif. Intell. Res*., **11**, 131–166.

Chan,T. *et al*. (2004) Finding biomarkers specific for early and late stages of lung cancer using sage data. In *International conference of 11th World Conference on Lung Cancer,* Barcelone, Spain, Supplement 2, 49, S120, P-032.

De la Torre,D. and Black,M.J. (2001) Robust principal component analysis for computer vision. In *Eighth International Conference on Computer Vision (ICCV01),* Vancouver, BC, Canada, Vol. 1, p. 362.

Dudoit,S. *et al*. (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. *J. Am. Stat. Assoc*., **97**, 77–87.

Furey,T.S. *et al*. (2000) Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, **16**, 906–914.

Golub,T.R. *et al*. (1999) Molecular classification of cancer: class discovery and class prediction bye gene expression monitoring. *Science*, **286**, 531–537.

Kadota,K. *et al*. (2003) Detecting outlying samples in microarray data: a critical assessment of the effect of outliers on sample classification. *Chem-Bio Inform. J*., **3**, 30–45.

Knorr,E.M. and Ng,R.T. (1998) Algorithms for mining distance-based outliers in large datasets. In *Proceedings 24th International Conference Very Large Data Bases, VLDB,* NY, USA, pp. 392–403.

Li,L. *et al*. (2001) Gene assessment and sample classification for gene expression data using a genetic algorithm/*k*-nearest neighbor method. *Comb. Chem. High Through. Scr*., **4**, 727–739.

Malossini,A., Blanzieri,E. and Ng,R.T. (2005) Assessment of SVM reliability for microarray data analysis. In *14th Dutch–Belgian Conference on Machine Learning, Benelearn 2005,* Enschede, Vol. WP05-03.

Muhlenbach,F., Lallich,S. and Zighed,D.A. (2004) Identifying and handling mislabelled instances. *J. Intell. Inform. Syst*., **22**, 89–109.

Ramaswamy,S. *et al*. (2003) A molecular signature of metastasis in primary solid tumors. *Nat. Genet*., **33**, 1–6.

Sanchez,J.S. *et al*. (2003) Analysis of new techniques to obtain quality training sets. *Patt. Recogn. Lett*., **24**, 1015–1022.

Vapnik,V. *et al*. (2002) Gene selection for cancer classification using support vector machine. *Mach. Learn*., **46**, 389–422.

Venkataraman,S. *et al*. (2004) Distinguishing mislabeled data from correctly labeled data in classifier design. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'04),* Boca Raton, FL, p. 668–672.

West,M. *et al*. (2001) Predicting the clinical status of human breast cancer by using gene expression profiles. *Proc. Natl Acad. Sci. USA*, **98**, 11462–11467.