Spot defects detection in cDNA microarray images

Authors:

Mónica G. Larese	**(1)**	**<larese@cifasis-conicet.gov.ar>**
Pablo M. Granitto	**(1)**	**<granitto@cifasis-conicet.gov.ar>**
Juan C. Gómez	**(2)**	**<jcgomez@fceia.unr.edu.ar>**

Affiliations:

(1) CIFASIS, French Argentine International Center for Information and Systems Sciences, UPCAM (France) / UNR-CONICET (Argentina)
Bv. 27 de Febrero 210 Bis, 2000 Rosario, Argentina
Tel.: +54-(0)341-4237248 Ext. 303
Fax: +54-(0)341-4237248 Ext. 301

(2) Laboratory for System Dynamics and Signal Processing,
FCEIA, Universidad Nacional de Rosario
Riobamba 245 Bis, 2000 Rosario, Argentina
Tel. / Fax: +54-(0)341-4808543 Ext. 106

Pattern Anal Applic manuscript No.
(will be inserted by the editor)

Spot defects detection in cDNA microarray images

Received: date / Accepted: date

Abstract Bad quality spots should be filtered out at early steps in microarray analysis to avoid noisy data. In this paper we implement quality control of individual spots from real microarray images. First of all, we consider the binary classification problem of detecting bad quality spots. We propose the use of ensemble algorithms to perform detection and obtain improved accuracies over previous studies in the literature. Next, we analyze the untackled problem of identifying specific spot defects. One spot may have several faults simultaneously (or none of them) yielding a multi-label classification problem. We propose several extra features in addition to those used for binary classification, and we use three different methods to perform the classification task: five independent binary classifiers, the recent Convex Multi-task Feature Learning (CMFL) algorithm and Convex Multi-task Independent Learning (CMIL). We analyze the Hamming loss and areas under the receiver operating characteristic curves (ROCs) to quantify the accuracies of the methods. We find that the three strategies achieve similar results leading to a successful identification of particular defects. Also, using a Random forests based analysis we show that the newly introduced features are highly relevant for this performance.

Keywords Microarray images · Quality control · Defects classification · Ensemble classifiers · Convex Multi-task Learning · Pattern recognition

1 Introduction

Spotted DNA microarrays are a high-throughput technology, which allows the analysis of thousands of genes simultaneously and study potential correlations among them [32, 27, 18, 2]. In a spotted microarray image thousands of spots represent the expression levels of the genes under study. However, these images present high variability in their quality due to intrinsic factors arising at the manufacturing process, such as the hybridization and printing steps, as well as the quality of the biological samples [13]. The bad quality of the images, and therefore, the spots, negatively affects the gene expression levels which are measured. Ideally, all the bad quality spots should be filtered at early steps in order to avoid wrong conclusions in the subsequent data analysis.

Most of the existing works in the literature concentrate on developing algorithms to efficiently locate and segment the spots in order to measure the expression levels [3, 11, 8, 14, 21, 7] (also a review of existing methods can be found in Bajcsy [5]), or on performing pattern recognition and data mining tasks to process and analyze the already extracted gene expression levels [22, 34, 15, 29] (the reader is referred to the work by Valafar [35] for a survey on this topic). In Blekas *et. al* [26] and Bozinov *et. al* [10], for example, the proposed segmentation algorithms are robust to the presence of artifacts, but no focus is placed on spot classification according to their quality nor on the identification of the faults.

Microarray analysis tools such as Spot [38], Scanalyze [17] and Genepix Pro [1] allow a human expert to manually flag out bad spots. This leads to a tedious and error prone procedure given the large number of available spots. These software packages also provide automatic flagging, but this is limited to computing several morphological and statistical measures which describe the spots (e.g. spot sizes, signal to noise ratio for individual channels, correlation between both channels). These measures are intended to be later combined in some way and thresholded in order to discard bad spots. However, these features are not used to find a model of bad quality spots nor to identify the specific defects.

Only a few works [30, 25, 9] deal with the particular problem of spots quality control by finding a model that discriminates bad from good spots.

In the work by Ruosaari and Hollmen [30], the authors extract spatial spot features and use them to train a Bayesian binary classifier, which separates the good quality spots from the bad quality ones. The spatial features consist of vertical and horizontal range, elongation, circularity, uniformity and Euclidean distance between the spot centers and the binary mask which is used to extract the spots.

An automatic quality control strategy based on Bayesian networks is proposed by Hautaniemi *et al.* [25], where a Gaussian 2D distribution is fitted to each spot in order to perform feature extraction. Bad quality spots are separated from good quality spots in this binary classification problem. Even though the experiments show that Bayesian networks are an effective tool for spot binary classification, it is necessary to define first their structure, describing the relations between the model components.

Bicego *et al.* [9] propose Support Vector Machines (SVMs) [37] to separate good from bad quality spots using the same features as in Hautaniemi *et al.* [25]. This method showed to improve the performance, and has the advantage that it does not require *a priori* knowledge about the data.

In this paper we address the problem of spot classification for microarray quality control. First of all, we perform bad/good quality spots discrimination by means of ensemble algorithms and using the same set of features proposed by Hautaniemi *et al.* [25] and used by Bicego *et al.* [9]. As stated above, this problem has already been tackled by these authors using several classifiers, including the recent and powerful SVMs. However, to the best of our knowledge, ensembles have not been used for this purpose yet. We compare our classification performances to those obtained in Hautaniemi *et al.* [25] and Bicego *et al.* [9], and show that ensembles can improve the accuracy of the discrimination.

Ensemble methods [24] are machine learning algorithms developed in the last decades, which leverage the power of multiple learners and combine their predictions in order to achieve better accuracy and more robustness than any of the single learners acting individually. The learners should be complementary to one another to take advantage of the method, because if they always agree there would not be any improvements over using the individual learners. Ensembles showed to be very competitive against the best state of the art learning algorithms, *i.e.* SVMs, achieving similar or even better performances (see e.g. Baluja *et al.* [6], where ensembles also show to be faster than SVMs, and Liu *et al.* [28]).

Regarding the bad/good quality classification problem, our goal is to implement an ensemble of binary classifiers able to predict the class label for each spot. We propose two powerful representatives of ensemble algorithms to solve this binary classification problem. These are Boosting and Random Forests.

Boosting is an iterative algorithm based on the idea that if several "weak" classifiers (simple classification rules that provide mis-classification errors slightly better than chance) are combined into an ensemble, the result will be a "strong" classifier with a highly improved performance [36, 31]. Perhaps the most common and simplest version of boosting is the AdaBoost algorithm [19], also called Discrete AdaBoost. Several variants have been proposed since Discrete AdaBoost appearance, including Real and Gentle AdaBoost [20].

Random forests [12] is an ensemble algorithm that creates a large set of uncorrelated trees and makes them vote for the final output. This voting over the decision of all the trees, called "bagging", reduces the high variability of the trees. Random forests compares favorably to boosting in performance, and is also faster and simpler to train and tune. Additionally, it has the advantage of incorporating an internal measure of input's relevance that can be used to assess the relative importance of each feature for the discrimination task.

Afterwards, we consider the problem of spot defects detection in order to identify the faults appearing in the spots. We propose a new set of additional features to represent the spots and use the recent algorithm of Convex Multi-task Learning [4] to perform classification. We use for this purpose real spots manually labeled by three human experts. We assume that one spot may suffer from several defects at the same time (or none of them) and that those faults are correlated, *i.e.*, the occurrence of one defect may affect the presence of others. For example, it is feasible to think that a spot of bad size (too small or too big) may also have a non-circular morphology, or a non-uniform pixel intensity distribution.

Our goal in this case is to detect all the defects in the spot, and this constitutes a multi-label classification problem. The identification of the faults and their correlation may be used as feedback to correct or improve the manufacturing process. Typical defects in cDNA microarray images are described in the literature [25], as well as the experimental factors that may cause their ocurrence. For example, as suggested by Hautaniemi *et al.* [25], big deviations in the spot sizes (which ideally should be aproximately all the same) may be caused by several issues, including the necessity for replacement of damaged needles.

The detection of defects may also let distinguish different degrees of reliability of the information provided by the spots. For example, spots suffering from bleeding are generally not reliable. However, some variations in the spot sizes are allowed and, unless the spots are too small or too big, this is usually not a drastic problem. The detection of the specific defects let calculate spot quality measures which can be incorporated into microarray analysis tools, such as the R package **limma** [33]. This package let the user weight the spots according to their quality in order to measure the reliability of the spot ratios for posterior analysis. The procedure proposed in this paper may be easily added to any microarray image analysis software to compute these weights. The scores obtained in the classification process may be used directly to rank the spots according to the different defects under consideration.

Multi-label classification tasks are much more complex than the simple binary classification, and thus the basic spot features previously used are not sufficient. We propose an additional set of features, which are extracted for the identification of defects and show to improve classification accuracy. In addition, we use the recent algorithm of Convex Multi-task Feature Learning (CMFL) [4] to perform multi-label classification. This algorithm is specifically designed for solving multi-task problems where the tasks are correlated, allowing to find an optimized shared representation of the features across the different classes. To the best of our knowledge, the identification of multiple spot defects by modeling the criteria of human experts via multi-task learning is a novel approach which has not been considered in the literature yet. We evaluate the performance in terms of the Hamming loss and the areas under the receiver operating characteristic curves (ROCs), obtaining good results.

The rest of the paper is organized as follows. We report in Section 2 the binary classification problem, where we separate spots into good and bad quality classes. We develop the defects identification problem in Section 3. Finally, we draw some conclusions in Section 4.

2 The “bad spots” detection problem

In this section we describe the dataset and explain the methods used to solve the good/bad quality classification problem. We also discuss the obtained results.

2.1 Dataset description

We use a publicly available dataset, which consists of spots extracted from two different microarray images [25]. A grid of one of these images is shown in Fig. 1. These spots were labeled by three human experts which have several years of experience dealing with microarray experiments. A total number of 320 spots (160 from each image) were assigned by the experts to four quality categories: bad, close to bad, close to good and good. The three experts labeling exactly coincides in 155 spots. In order to perform binary classification, we grouped the previously mentioned four quality categories into good quality (by joining “good” and “close to good” sets) and bad quality (the union of “bad” and “close to bad” sets). Using these settings, we found 97 out of the 155 spots to belong to the good quality class and 58 to the bad quality class. These same settings were used by Hautaniemi *et al.* [25] and Bicego *et al.* [9]. The microarray images, experts labeling and additional information about the dataset are publicly available[1].

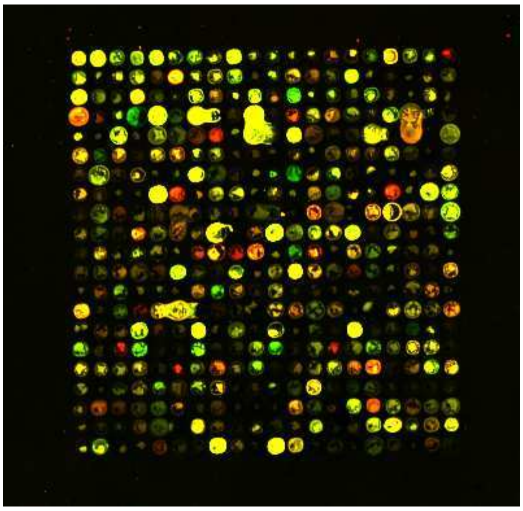


Fig. 1 One of the microarray grids containing the spots to be classified.

2.2 Basic feature set

Hautaniemi *et al.* [25] proposed to compute seven features per spot for each channel (Cy3 and Cy5) in order to classify spots into good and bad classes, giving rise to a 14-component feature vector. We computed the seven features as follows.

First of all, we fitted a 2D Gaussian surface to every spot using a standard non-linear least squares procedure over a 15×15 pixel grid [25]. This function is the estimation of the spot intensity distribution across the spot pixels. The 2D function is defined as [25]

$$f(\mathbf{x}, A, B, \bar{\mathbf{x}}, \sigma_x, \sigma_y, \phi) = Ae^{-(\mathbf{x}-\bar{\mathbf{x}})^T \mathbf{S}(\mathbf{x}-\bar{\mathbf{x}})} + B \tag{1}$$

where $\mathbf{x} = [x, y]^T \in \mathbb{R}^2$ is the pixel coordinates vector, $\bar{\mathbf{x}} \in \mathbb{R}^2$ is the Gaussian mean, and $\mathbf{S} = R_\phi^T \operatorname{diag}\left(\sigma_x^{-2}, \sigma_y^{-2}\right) R_\phi$ is the inverse covariance matrix, where σ_x and σ_y are the x and y standard deviations, respectively, and R_ϕ is the rotation matrix with rotation angle ϕ. Parameters A and B are the foreground and background intensities of the spot, respectively. In practice, the size of the pixel grid should not be critical, given that it is high enough as to clearly contain the spot.

[1] `http://www.cs.tut.fi/TICSP/SpotQuality/`

We extracted the seven features from this 2D Gaussian function in the following way (the reader is referred to [25] for more details on how to compute these features):

- Spot intensity: parameter A.
- Background intensity: parameter B.
- Alignment error: the distance between $\bar{\mathbf{x}}$ and the center of the spot bounding box.
- Roundness: the σ_x/σ_y ratio.
- Spot size: the $\sigma_x\sigma_y$ product.
- Background noise: the root mean square error between the spot and the fitted 2D Gaussian function.
- Bleeding: the number of pixels of the spot which fall outside the fitted Gaussian.

The final feature vector is composed by the aforementioned continuous features for Cy3 and Cy5 channels. In contrast to previous works [25, 9], we do not apply any discretization procedures nor feature selection algorithms on the feature set in order to perform binary classification, since it did not provide any improvements in our case.

2.3 Ensemble algorithms

For the sake of completeness, next we briefly describe ensemble algorithms [24].

2.3.1 Boosting

Boosting classifiers are based on the idea that if many "weak" classifiers (slightly better than chance) are combined into a "strong" classifier, the overall performance will be highly improved [36, 31]. In this paper we considered three different boosting algorithms, namely Discrete, Real and Gentle Boost, which we describe below.

Let $D = \{(\mathbf{x}_i, y_i)\}$, with $i = \{1, ..., n\}$, be a training dataset of n pairs of feature vectors $\mathbf{x}_i \in \mathbb{R}^p$ and class labels $y_i \in \{-1, 1\}$. Discrete AdaBoost [19] creates a sequence of weak classifiers $(f_m(\mathbf{x}_i))$ aimed at discriminating the training observations. Initially, all the observations are assigned a unique weight $w_{i,m}$. This distribution of weights is modified along with the $m = \{1, ..., M\}$ iterations (rounds), *i.e.*, observations which are badly classified (more difficult to learn) are given higher weights. The algorithm attempts to find an optimum classifier at each round. Each weak classifier is weighted according to its performance on the current distribution of weights on the observations. At the end, the final strong classifier $F(\mathbf{x}_i)$ is the weighted linear combination of the weak classifiers, as shown in Eq. (2).

$$F(\mathbf{x}_i) = \text{sign}\left(\sum_{m=1}^{M} w_{i,m} f_m(\mathbf{x}_i)\right) \tag{2}$$

The output of Discrete AdaBoost at each iteration is a discrete value corresponding to the predicted class label for each observation. The efficiency of the algorithm may be improved by computing class probabilities instead of discrete labels. These class probabilities are then converted to the real scale and used to update the weight distribution for the observations at each iteration. This improved algorithm is named Real AdaBoost [20]. Both Discrete and Real AdaBoost minimize the expectation of the so-called "exponential loss", defined as $e^{-y_i F(\mathbf{x}_i)}$. Gentle AdaBoost [20] is an algorithm very similar to Real AdaBoost, but uses a sequence of Newton steps to optimize the expectation of the exponential loss. Even though the classification results are very similar for both methods, this feature makes Gentle AdaBoost numerically superior to Real AdaBoost.

2.3.2 Random forests

Random forests [12] is a recent kind of ensemble algorithm, where the individual classifiers are a set of de-correlated trees. They perform similarly or even better than boosting in some situations, and are faster too.

The algorithm works by building a collection of unpruned trees from B random samples with replacement (bootstrap versions) of the original training dataset. For each random forest tree f_b, a random sample of $m \leq p$ variables is selected to split the data at each node and grow the decision tree. The final classification result $F(\mathbf{x}_i)$ is the class corresponding to the majority vote of the ensemble of trees:

$$F(\mathbf{x}_i) = \text{majority vote } \{f_b(\mathbf{x}_i)\}_{b=1}^{B} \tag{3}$$

As we mentioned before, Random forests incorporates a mechanism for the estimation of the importance of input variables. As explained by Breiman [12], after the model was trained, features are shuffled (*i.e.* their values are randomly permuted between all cases in the dataset) one at a time. Then, an out-of-bag estimation of the prediction error is made on this "shuffled" dataset. Intuitively, a feature that is irrelevant to the model will not change the prediction performance when altered in this way. On the other hand, if the model made strong use of a given feature, altering its values will lead to an important decrease in performance. The relative loss in performance between the "original" dataset and the "shuffled" dataset is therefore related to the relative relevance of the feature affected by the process.

2.4 Experimental results

We performed classification with the three versions of boosting classifiers described in Subsection 2.3.1, namely Discrete, Real and Gentle AdaBoost, resorting to the R package

Table 1 Test errors for the different classifiers using LOOCV.

Classification Algorithm	**Accuracy**
B-Course (subjective) [25]	96.8%
Pair-wise NB (subjective) [25]	95.5%
NB (subjective) [25]	95.5%
NB (uniform) [25]	94.8%
Decision Tree [25]	91.6%
Neural Networks [25]	90.3%
SVM (Linear)[9]	96.1%
SVM (Polynomial)[9]	94.2%
SVM (Gaussian RBF)[9]	97.4%
Discrete AdaBoost (Proposed approach)	96.8%
Real AdaBoost (Proposed approach)	97.4%
Gentle AdaBoost (Proposed approach)	98.1%
Random Forests (Proposed Approach)	**98.7**%

Table 2 Mean test error and standard error (S_E) for random subsampling and 20 times 5-fold cross validation.

Classification algorithm	**Test error (mean$\pm S_E$)** **Random subsampling**	 **20 times 5-fold CV**
SVM (Gaussian RBF kernel)	0.0368 ± 0.0034	0.0300 ± 0.0034
Discrete AdaBoost	0.0342 ± 0.0030	0.0303 ± 0.0030
Real AdaBoost	0.0342 ± 0.0030	0.0310 ± 0.0033
Gentle AdaBoost	0.0330 ± 0.0031	0.0310 ± 0.0032
Random Forest	**0.0253 ± 0.0026**	**0.0235 ± 0.0028**

ada [16], using stumps as weak learners. We implemented the random forests algorithm described in Subsection 2.3.2 via the R package **randomForest** [2]. We used 1000 rounds for each of the boosting methods and 500 trees for random forests. We computed the features as detailed in Subsection 2.2, and considered a total number of 155 spots for this binary problem, which are the spots with unanimous labeling by the three experts. Following the same methodology employed in previous works in the literature [25, 9], we used leave-one-out cross-validation (LOOCV) to assess the performance, and we compared the generalization error to those obtained in these previous works.

We show in Table 1 the accuracies of the four ensemble methods against previous results existing in the literature. We obtained very good performances for all the ensemble algorithms. From Table 1 we can see that Discrete AdaBoost resulted in the same accuracy as B-Course (subjective) [25], and Real AdaBoost provided the same improvement as SVM with Gaussian RBF kernel [9]. Gentle AdaBoost reduced the mis-classifications from four to three out of the 155 spots. However, we found Random forests to be the most accurate algorithm (highlighted in gray) with only two mis-classified spots.

As LOOCV is an unstable procedure, usually resampling is implemented instead. We also computed the mean and its standard error for the generalization error by running the different classifiers 100 times, using each time a random partitioning of the dataset with 75% of the spots as training set and leaving the remaining 25% for testing purposes. We show these results in Table 2. This form of computing the error is more pessimistic than LOOCV, since the training set becomes smaller. We computed the error for the four ensemble algorithms and SVMs (which showed to have the best performance, according to Table 1, among the previously proposed classifiers). We performed a similar experiment by running 20 times 5-fold cross-validation. We also show these results in Table 2. In all the cases, Random forests obtain the best performance with the lowest standard error.

We can compare the errors in Table 2 to the accuracies in Table 1, which show to be consistent (taking into account that accuracies are defined as “1-error rate”). The three error measures behave as expected, indicating that Random forests provide the lowest errors in all the cases.

For the classifiers shown in Table 2, we computed the areas under the Receiver-Operating-Characteristic (ROC) curves, obtaining values greater than 0.97 in all the cases. This fact indicates that all the algorithms under consideration provide good accuracies with a very low level of random discrimination, and that the binary problem seems to be a relatively easy classification problem.

[2] `http://stat-www.berkeley.edu/users/breiman/RandomForests`.

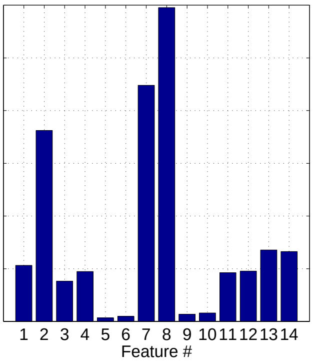


Fig. 2 Feature relative importance for the binary problem.

In Fig. 2 we show the relative importance of the 14 features measured by the Random forests classifier. The features are ordered in pairs corresponding to the red and green channels for intensity, background intensity, roundness, spot size, alignment error, background noise and bleeding. For example, bins 3 and 4 in Fig. 2 show the relative importance of both channels for the background intensity feature. For the binary problem, it is clear that the most relevant features are the spot size computed on both channels (bins 7 and 8). Additionally, the intensity of the green channel seems to be also important. In contrast, the less relevant features are the roundness (bins 5 and 6) and the spot alignment error (bins 9 and 10). It is interesting to note that in only one out of the 7 pairs of features there are marked differences between the red and green channels.

3 The “defects identification” problem

We describe in this section the dataset and the procedures followed to detect the faults present in the spots. We describe the different kinds of faults under consideration as well as the representation features that we used. We also discuss the experimental results.

3.1 Dataset description

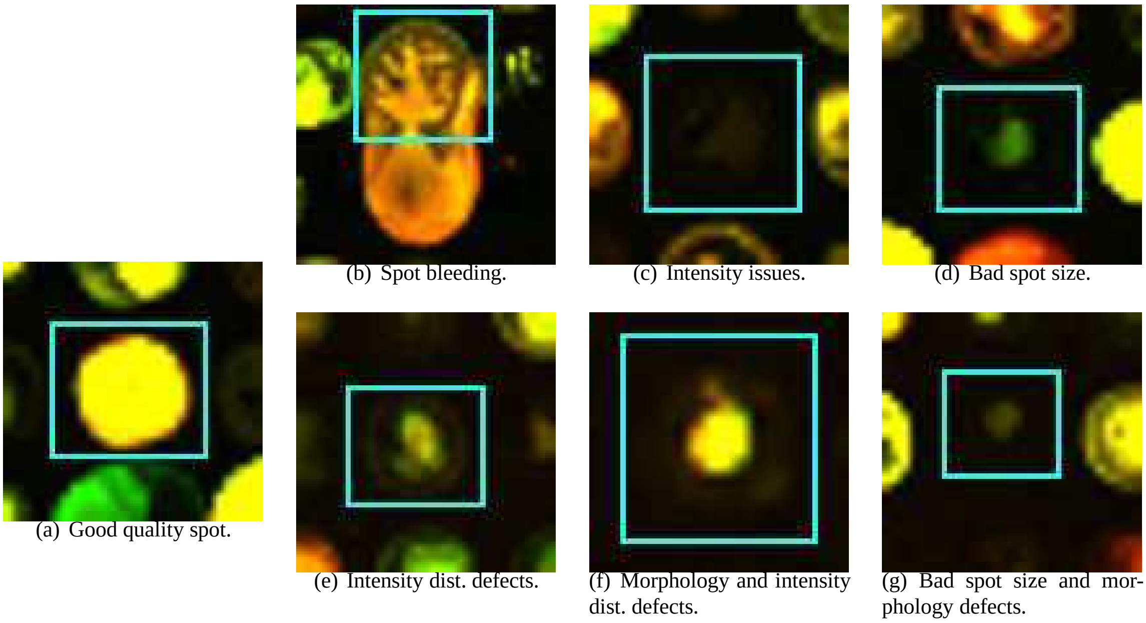
For the multi-label classification problem, we considered the totality of the 320 spots from two different microarray images belonging to the dataset described in Subsection 2.1.

The three experts who manually classified the spots into the bad/good quality classes also labeled them according to their defects. The six types of faults that we take into account in the experiment are

- Spot bleeding: it occurs when one spot overlays one or more of its neighbors.
- Background defects: e.g. non-specific hybridizations or noise present in the background.
- Bad spot size: all the spots are supposed to have approximately the same size (measured in pixels). This problem concerns spots whose size deviates too much from the rest, for instance, due to problems in the printing needles.
- Morphology defects: the spots whose morphology deviates too much from a circle fall in this category.
- Pixel intensity distribution defects: the foreground intensities should be uniform for good quality spots. However, non-specific hybridizations or uneven distribution of the DNA samples may cause this defect to appear, as in the case of donuts or holey spots.
- Intensity issues: they happen when the foreground signal is too weak due to genes which express at very low levels, incomplete hybridization or low sensitivity of the scanner. It causes difficulties at the segmentation step since there is no good contrast of the spot against the background.

We show some spot examples in Fig. 3. In Table 3 we report the number of spots in each category, according to the judgment of each expert. First, we report the number of spots according to the independent labeling of each expert, *i.e.*, it is enough for one spot that only one expert considers it as belonging to a faulty class. Next, we inform the number of spots for which two experts agreed. Finally, we detail the agreement of the three experts. It is worth mentioning that there are no instances of spots suffering from background problems and, in consequence, we excluded this class in the analysis.

From Table 3 it is evident that the conclusions drawn by the experts are very subjective, and that the criteria they used to classify spots according to their defects are not homogeneous. We can assume that the experts should agree in cases where defects are clearly evident, and that they should produce differences in less marked situations. There are two possible causes for those differences. On one side, it is possible that all experts completely agree in the definition of the defects, and that in some cases any of them simply missed to mark a defect in a spot. On the other side, it could be the case that each expert is consistent in his own classification but has a different opinion on which is the particular defect present in a spot. This situation would produce a matrix of co-occurrence of defects with high values at non-diagonal positions (as the same defect will be assigned two different labels). In Table 4 we show the corresponding matrix for our dataset, where several high non-diagonal values are present. Of course, it is expected that some spots really suffer from more than one fault, but the high disagreement observed in Table 3 together with the high co-occurrence of faults in Table 4 points clearly to a set of experts that, in some cases, have different concepts of each defect.



(a) Good quality spot.

(b) Spot bleeding.

(c) Intensity issues.

(d) Bad spot size.

(e) Intensity dist. defects.

(f) Morphology and intensity dist. defects.

(g) Bad spot size and morphology defects.

Fig. 3 Some examples of spots having good quality or different types of defects.

There are at least two ways to face this problem. On one side, we can choose to reduce the dataset considering only the agreement of two or three experts. Unfortunately, this results in low populated classes, hardening the modeling process. Oppositely, we choose to use the dataset as it is, keeping the opinion of all experts. This is a more difficult problem than the one we would face if the experts would agree in all cases, but if we can solve this problem with high accuracy, it is expected that the same can be done with the easier problem of coherent experts.

3.2 Additional feature set

We analyzed the seven basic features proposed by Hautaniemi *et al.* [25] and previously used to deal with the binary classification problem for the multi-label problem too. We already described them in Subsection 2.2. They gave rise to a 14-component vector for each spot (7 features per channel). However, they did not perform very well for the multi-label problem, as we show in Subsection 3.4. For this reason, we propose in this paper the following additional spot features. They are computed on the grayscale image for each spot.

- Intensity projection profiles of the spot image f. These profiles are obtained by summing up the intensities along the rows r (vertical sum profile S_v) and along the columns c (horizontal sum profile S_h). This procedure generates two vectors corresponding to the intensity summation in each direction:

  $$S_v(c) = \sum_r f(r,c)$$

  and

  $$S_h(r) = \sum_c f(r,c).$$

- Euler number ε: the number of connected components nc in the image region minus the number of holes nh, *i.e.*: $\varepsilon = nc - nh$.
- Eccentricity ξ: eccentricity of the ellipse, which has the same second moments as the spot. Let d_1 be the distance between the foci of the ellipse, and d_2 the length of its major axis. The eccentricity ξ is equal to $\xi = d_1/d_2$. The values of ξ ranges from 0 (a circle) to 1 (a line segment).
- Spot perimeter P: computed as

  $$P = \sum_r \sum_c I(r,c),$$

  where I is a binary image with 1s in the perimeter pixels. The perimeter pixels are those having at least one neighbor pixel equal to 0.
- Texture descriptors [23]:
  - Mean of the intensity histogram (first order statistical moment, average gray level):

    $$\mu = \sum_{i=0}^{L-1} z_i p(z_i),$$

    where L is the number of possible gray levels and $p(z)$ is the probability distribution of intensities z_i.

Table 3 Class distributions.

	Spot bleeding	**BKG defects**	**Bad spot size**	**Morphology defects**	**Int. distrib. defects**	**Intensity issues**
3 expert independent conclusions (no agreement required)						
Number of spots	12	0	93	73	98	130
Probability	0.0375	0	0.2906	0.2281	0.3063	0.4062
2 experts agreement						
Number of spots	9	0	66	25	28	35
Probability	0.0281	0	0.2062	0.0781	0.0875	0.1094
3 experts agreement						
Number of spots	6	0	30	2	4	10
Probability	0.0187	0	0.0938	0.0063	0.0125	0.0312

Table 4 Contingency table for the five classes of defects.

Classes	**Spot bleeding**	**Bad spot size**	**Morphology defects**	**Int. distrib. defects**	**Intensity issues**
Spot bleeding	**12**	0	0	2	3
Bad spot size	0	**93**	51	25	66
Morphology defects	0	51	**73**	34	45
Int. distrib. defects	2	25	34	**98**	50
Intensity issues	3	66	45	50	**130**

- Standard deviation of the intensity histogram (second order statistical moment, measure of contrast), defined as:

$$\sigma = \sqrt{\sum_{i=0}^{L-1} (z_i - \mu)^2 p(z_i)}.$$

- Skewness of the intensity histogram (third order statistical moment, asymmetry about the mean):

$$s = \sum_{i=0}^{L-1} (z_i - \mu)^3 p(z_i).$$

- Smoothness: $R = 1 - 1/(1+\sigma^2)$. This measure ranges from 0 (spot with constant intensities) to 1.
- Uniformity (also energy):

$$U = \sum_{i=0}^{L-1} p^2(z_i).$$

  This measure is maximum when the intensities are uniformly distributed.
- Average entropy as a measure of randomness:

$$E = \sum_{i=0}^{L-1} p(z_i) \log_2 p(z_i).$$

As we mentioned above, we do not compute the previously described features on the red and green channels separately, but on the grayscale image of the spot (typically obtained as the average between the red and green channels). In particular, for the Euler number, eccentricity and spot perimeter we used binary images obtained using a typical fixed threshold at a half of the range of the grayscale image. Afterwards, we concatenate all the features into a single vector with a total number of 49 components.

In this work, the intensity profiles are computed over 13 × 13 spot masks. This number was selected because it is the standard size for good quality spots in this dataset, and it showed to perform well. In general, the number of bins used to compute the intensity profiles should not be critical, as we are simply estimating a histogram of the spatial distribution. As usually, the number of bins should be high enough as to see some detail, but not too big as to introduce noise in the problem. Also, this fixed number should not be a problem with test spots with a different size, as the images can be easily sampled at the correct resolution.

3.3 Convex Multi-task Feature Learning (CMFL)

Learning low-dimensional features shared across different classes/tasks has shown in the literature to improve the performance against learning the individual classes/tasks. Argyriou *et al.* [4] proposed a method which learns a shared low-dimensional representation of the features among the different tasks by regularizing within the tasks while keeping them coupled to each other.

Let $D = \{(\mathbf{x}_i, y_i)\}$ be a labeled dataset with $i = 1,...,m$ input/output observations. Every single vector $\mathbf{x}_i \in \mathbb{R}^d$ is the input vector with associated label $y_i \in \{\pm 1\}$. The supervised learning process can be formulated as a single task classification problem, where the labels are determined by a margin classifier $f : \mathbb{R}^d \to \mathbb{R}$, such that

$$f(\mathbf{x}_i) = \sum_{j=1}^{d} a_j h_j(\mathbf{x}_i), \tag{4}$$

where $h_j : \mathbb{R}^d \to \mathbb{R}$ are the features and $a_j \in \mathbb{R}$ are the regression parameters.

The single class/task problem can be extended to a multi-class/task problem by introducing the vectors $\mathbf{y}_i \in \mathbb{R}^T$ containing the labels for T different classes/tasks. In the general case, the input observations $\mathbf{x}_{ti}$ (with $t = 1,...,T$) may be different for each class/task in the multi-class/task situation.

The margin classifiers for each class, f_t, are expected to be related to each other and share a small set of features, and only a few features are expected to have non-zero coefficients across all the classes/tasks.

Let assume now that the features are linear, *i.e.*, $h_j(\mathbf{x}_{ti}) = \langle \mathbf{u}_j, \mathbf{x}_{ti} \rangle$, with vectors $\mathbf{u}_j \in \mathbb{R}^d$ and orthonormal. Let U be the $d \times d$ matrix with columns formed by the vectors $\mathbf{u}_j$. Assume that the functions f_t are also linear, *i.e.*, $f_t(\mathbf{x}_{ti}) = \langle \mathbf{w}_t, \mathbf{x}_{ti} \rangle$, with $\mathbf{w}_t = \sum_{j=1}^{d} a_{jt} \mathbf{u}_j$.
Extensions to nonlinear functions are possible, e.g., by using kernels, but they are outside the scope of this work.

Denote by W the $d \times T$ matrix with vectors $\mathbf{w_t}$ as columns, and by A the $d \times T$ matrix with entries a_{jt}. Then $W = UA$. As it is expected to find a low-dimensional set of features shared by all the classes, the matrix A has many rows equal to zero and the corresponding features (columns of U) will be discarded to represent the task parameters (columns of W). Matrix W is then a low rank matrix.

The solution to the learning problem then reduces to the computation of the feature vectors $\mathbf{u}_j$ and the parameters a_{jt} which minimize the unconstrained problem

$$\min\{\mathscr{E}(A,U) : U \in \mathbf{O}^d, A \in \mathbb{R}^{d\times T},\} \tag{5}$$

where

$$\mathscr{E}(A,U) = \sum_{t=1}^{T}\sum_{i=1}^{m} \mathscr{L}(y_{ti}\langle \mathbf{a}_t, U^\top \mathbf{x}_{ti}\rangle) + \gamma ||A||_{2,1}^2. \tag{6}$$

In Eq. (5), $\mathbf{O}^d$ is the set of $d \times d$ orthonormal matrices, and in Eq. (6), $\gamma > 0$ is the regularization parameter. The first term in this equation is the average of the empirical error across the tasks, while the second term is the regularizer which penalizes the (2,1)-norm of the matrix A. This norm is the responsible for combining the tasks and the selection of common features across them. The number of non-zero elements of $b(A)$ represent the importance of each derived feature across the tasks, also favoring uniformity across the tasks.

3.4 Experimental results for the “defects identification” problem

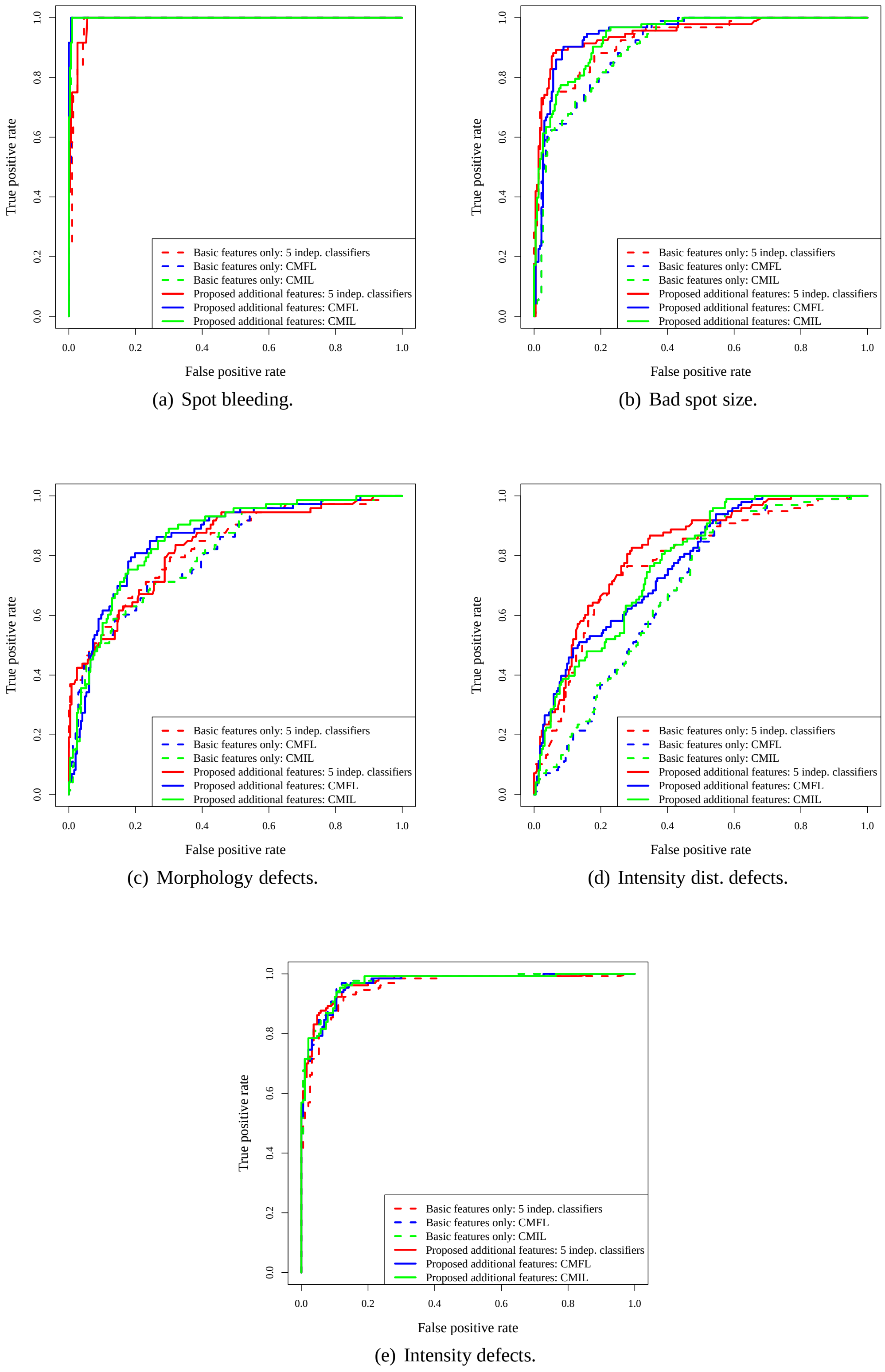
As we showed in Subsection 2.4, very good accuracies can easily be obtained for the detection of faulty spots using a basic set of features and standard classification algorithms. However, determination of the specific defects which affect the spots becomes a more complicated task.

For the defects classification problem, which is a multi-label task, we considered only the original set of features described in Subsection 2.2 at first. However, the results were not very good and these features seemed to be not good enough to extract the most relevant characteristics of the different defects when used alone. In order to improve accuracy, we additionally computed the features described in Subsection 3.2.

The ensemble algorithms described in Subsection 2.3 are not multi-class nor multi-label. As Random forests reached the best performance for the bad/good spots problem, we selected this classification algorithm to implement 5 independent classifiers aimed at detecting the presence/ absence of each defect separately.

We compared this approach to the very recent algorithm of Convex Multi-task Feature Learning (CMFL) [4] described in Subsection 3.3, which is able to find a shared representation of features for all the classes and perform multi-label classification. Additionally, we computed the results obtained by Convex Multi-task Independent Learning (CMIL, similar to Convex Multi-task Feature Learning but with no coupling across the classes, *i.e.*, using $||W||_2$ regularization). We tested all the algorithms using leave-one-out cross validation.

We computed the Hamming loss to measure the error in class predictions, since in most cases it provides a simple interpretation of the classifiers accuracy. It is calculated as the percentage of erroneously predicted labels for all the classes. In Table 5 we report the Hamming loss obtained by each classification algorithm with each different set of features. From these results we can see that using 5 independent classifiers provides the lowest error. The addition of the set of features described in Subsection 3.2 improves the accuracy of all the classification algorithms.



(a) Spot bleeding.

(b) Bad spot size.

(c) Morphology defects.

(d) Intensity dist. defects.

(e) Intensity defects.

Fig. 4 ROC curves for each defect class using different feature sets and classification algorithms.

Table 5 Hamming loss for the different classification algorithms and different feature sets.

	Hamming loss (%)
Basic features only and 5 IC	13.63
Basic features only and CMFL	27.69
Basic features only and CMIL	27.81
Proposed additional features and 5 IC	**11.94**
Proposed additional features and CMFL	25.63
Proposed additional features and CMIL	24.81

Table 6 Areas under the ROC curves (highlighted in gray the best results).

	Spot bleeding	
	Basic features only	**Proposed additional features**
5 IC	0.985	0.989
CMFL	0.996	**0.999**
CMIL	0.997	0.998
	Bad spot size	
	Basic features only	**Proposed additional features**
5 IC	0.930	0.948
CMFL	0.904	**0.949**
CMIL	0.901	0.939
	Morphology defects	
	Basic features only	**Proposed additional features**
5 IC	0.828	0.830
CMFL	0.807	0.855
CMIL	0.806	**0.858**
	Int. distribution defects	
	Basic features only	**Proposed additional features**
5 IC	0.780	**0.819**
CMFL	0.686	0.774
CMIL	0.688	0.777
	Int. defects	
	Basic features only	**Proposed additional features**
5 IC.	0.954	0.969
CMFL	**0.972**	0.968
CMIL	**0.972**	0.970

Even though the independent classifiers show the lowest Hamming Loss, they do not provide a useful decision function in this case, as a detailed analysis show that they are simply choosing to predict all spots as defectless (which in fact produces the lowest error rate, as faulty spots are clearly minoritary). A deeper analysis of the performance of the different methods can be obtained resorting to the Receiver-Operating-Characteristic (ROC) curves (Fig. 4), which show the relation between the true positive rate (TPR) and the false positive rate (FPR) for each class. From the figures it is evident that the three classification methods are, in fact, very similar. Evidently, the Hamming loss is acting on a portion of the ROCs where the curve of the independent classifiers exceeds the curve of the Convex Multi-task algorithms.

As we can appreciate from Figures 4(c) and 4(d), the morphology and intensity distribution defects are the most difficult to detect, since the ROC curves drawn after using only the basic set of features are closer to the identity curve corresponding to random classification. The areas under the ROCs (AUCs) that we show in Table 6 also confirm this. These two classes are also the ones which obtain the greatest improvements after adding the proposed features.

When using only the basic features for the defects identification problem, the 5 independent classifiers get the highest AUCs for the classes involving bad spot sizes, morphology and intensity distribution defects. CMIL and CMFL obtain the best performance to detect intensity defects and CMFL to identify spot bleeding. After adding the proposed features, all the AUCs are improved, except for a very slight reduction in the performance of the CMFL and CMIL algorithms for the class corresponding to intensity defects. With this feature set CMFL shows the best performance for the spot bleeding and bad spot sizes problems, CMIL for intensity and morphology defects and the 5 independent classifiers for the remaining problem. There is not a clear winner among the three methods analyzed in this work. Overall, considering all classes and classifiers, we achieve a high accuracy in this defects identification problem, in all cases with an AUC of over 0.81.

As we discussed in Section 3.1, we considered for this analysis the judgments of the three experts without taking into account the lack of consensus among them. In a short experiment, we also applied the 5 independent classifiers to the dataset produced by considering the agreement between two experts. In Table 7 we show the corresponding AUCs. There is a clear global improvement in the accuracy of the method, which indicates that this is an easier problem to solve, as we argued before.

In Fig. 5 we show, as an image, the coefficients of the matrix *A* resulting from the multi-label classification of the spots into the five classes using CMFL and the enlarged set of features. This matrix has five non-zero rows of coefficients (see Subsection 3.3). This means that the algorithm is able to find a shared representation of 5 features, obtained from the initial 49, among all the classes of defects. This shows that the intrinsic dimension of the problem is, in fact, low.

The relative importance of each feature for the five classes of defects using the 5 independent classifiers and the extended set of features is depicted in Fig. 6. From this figure,

Table 7 Areas under the ROC curves obtained by the 5 IC using the agreement of 2 experts and LOOCV.

Spot bleeding	*Bad spot size*	*Morphology defects*	*Int. distribution defects*	*Int. defects*
0.984	0.973	0.902	0.828	0.922

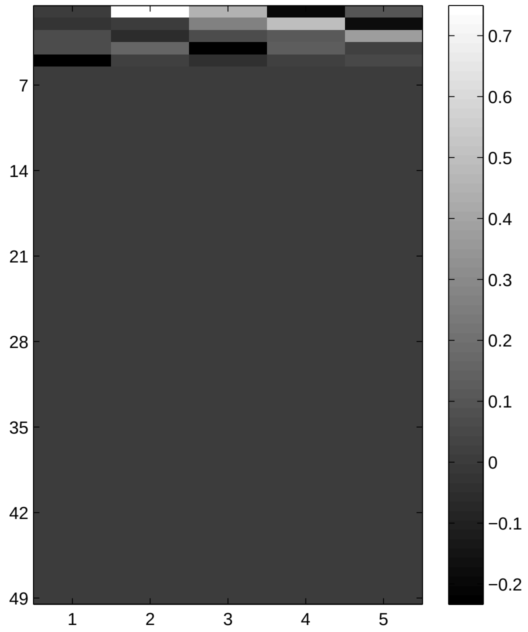


Fig. 5 Matrix *A* resulting from the CMFL algorithm, showing only 5 shared features among the classes.

it is evident that the bleeding features, computed on the red and green channels (bins 13 and 14 in group (A)) are clearly the most relevant to detect the bleeding class, as well as the intensity at the spot borders (first intensity profile in group (C)). A scatter plot of these relevant features (not shown) suggests that the defects are always associated with high values of this set of variables. The features from group (B) seem to be not relevant in this case.

Concerning the spot size defect, the most relevant features correspond to the spot size (bins 7 and 8 in (A)) for the red and green channels, the intensity profiles out of the center in (C) and (D), and the mean of the intensity histogram (bin 19 in (B)). In all cases, low values of the variables are associated with the defect. It is interesting to note that the same holds for the morphology defects detection, except for the mean of the intensity histogram. In the last case, none of the features in (B) seem to be of much importance.

According to the fourth row in Fig. 5, the most important features to detect the intensity spatial distribution defect are the perimeter (bin 23 in (B)), the mean of the intensity histogram, and the intensity profiles. In this dataset, the perimeter has low values for spots suffering from this defect.

Finally, to detect the intensity defects the features which show to be more relevant are the skewness, the mean and the standard deviation of the intensity histogram (bins 15, 19 and 20 in (B)), and the intensity profiles ((C) and (D)), all showing low values in the presence of the defect.

4 Conclusions

In this paper, we analyzed two different problems related to spot quality control. First of all, we considered the binary problem of separating good from bad quality spots by means of ensemble classifiers. We proposed to implement four ensemble algorithms to perform classification, namely Discrete, Real and Gentle AdaBoost, and Random forests. Random forests showed to perform better than the other algorithms already proposed in the recent literature. We computed basic features on the spots yielding very good accuracies.

The second problem was much more complex. It required the detection of five specific types of failures affecting the spots. As Random forests showed to be the most accurate classifier for the binary classification task, we tried it as the basis for five independent classifiers. The set of basic features suggested in previous works showed to be not good enough for this problem. Thus we proposed an additional set of features, which clearly improved the performance of all methods, showing that it is possible to identify the individual defects with high accuracy. Additionally, a Random forests analysis of features importance confirms that the new features are highly relevant for the discriminant models. Finally, we compared the independent classifiers to the recent algorithms of Convex Multi-task Learning, specifically CMFL and CMIL, finding that the overall performance of the three methods is equivalent on this problem.

As we discussed in the introduction, the results we obtained in this work allow to compute spot quality measures which can be later used in microarray analysis software. Some microarray analysis packages, e.g. the R package **limma** [33], let the user weight the spots according to their quality and use those weights to measure the reliability of the spot ratios for further analysis. The results obtained in the classification process described in this paper may allow to rank the spots according to the presence of the different defects.

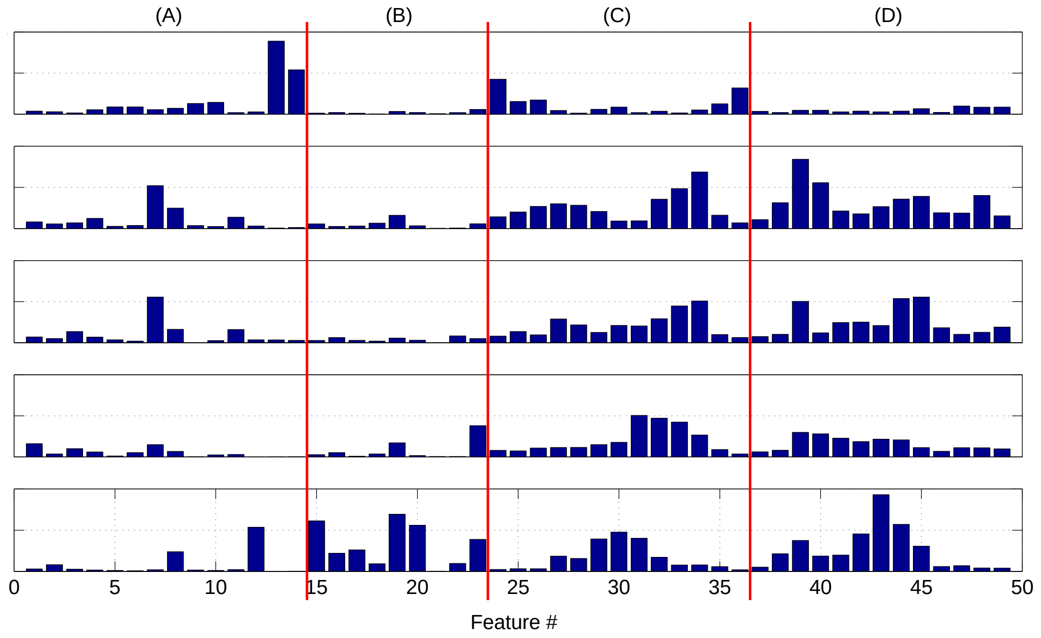


Fig. 6 Feature relative importance for the five defects using 5 IC. From top to bottom: bleeding, spot size defects, morphology defects, intensity distribution problems and intensity defects. From left to right: (A) Hautaniemi [25] features as in Fig. 2; (B) skewness, smoothness, uniformity, average entropy, mean and standard deviation of the intensity histogram, Euler number, eccentricity and perimeter; (C) and (D) intensity profiles.

A word is needed about the validity of our results. As in all previous works, we are assuming that our dataset is a fair sample of all existing microarray images, and in this context we show that our methods have very good generalization capabilities. Of course, in order to evaluate the real validity of our method and all previously published works much more extended studies are needed, comprising hundreds of images from different types of microarrays, experimental conditions, etc. However, one can expect that our results and also those available in previous studies will generalize well to new microarrays, as there are not methodological differences among images from different microarrays.

Acknowledgements

References

1. Axon GenePix Pro 7.1. `http://www.moleculardevices.com`
2. Alizadeh AA, Eisen MB, Davis EE, Ma C, Lossos IS, Rosenwald A, et al (2000) Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403(6769):503–511
3. Angulo J, Serra J (2003) Automatic analysis of DNA microarray images using mathematical morphology. Bioinformatics 19(5):553–562
4. Argyriou A, Evgeniou T, Pontil M (2008) Convex multi-task feature learning. Mach Learn, Special Issue on Inductive Transfer Learning 73(3):243–272
5. Bajcsy P (2006) An overview of DNA microarray grid alignment and foreground separation approaches. EURASIP J Appl Sig P Article ID 80163:1–13
6. Baluja S, Rowley HA (2007) Boosting sex identification performance. Int J Comput Vision 71(1):111–119
7. Bariamis D, Maroulis D, Iakovidis D (2009) Unsupervised SVM-based gridding for DNA microarray images. Comput Med Imaging Graph
8. Bengtsson A, Bengtsson H (2006) Microarray image analysis: background estimation using quantile and morphological filters. BMC Bioinf 7(96):1–15
9. Bicego M, Martínez MDR, Murino V (2005) A supervised data-driven approach for microarray spot quality classification. Pattern Anal Applic 8:181–187
10. Bozinov D, Rahnenführer J (2002) Unsupervised technique for robust target separation and analysis of DNA microarray spots through adaptive pixel clustering. Bioinformatics 18(5):747–756
11. Brändle N, Bischof H, Lapp H (2003) Robust DNA microarray image analysis. Mach Vision Appl 15(1):11–28
12. Breiman L (2001) Random forests. Mach Learn 45:5–32

13. Brown CS, Goodwin PC, Sorger PK (2001) Image metrics in the statistical analysis of DNA microarray data. Proc Nat Acad Sci USA 98(16):8944–8949
14. Chen TB, Lu HHS, Lee YS, Lan HJ (2008) Segmentation of cDNA microarray images by kernel density estimation. J Biomed Inf 41:1021–1027
15. Chopra P, Kang J, Yang J, Cho HJ, Kim HS, Lee MG (2008) Microarray data mining using landmark gene-guided clustering. BMC Bioinf 9(92):1–13
16. Culp M, Johnson K, Michailides G (2006) ada: An R Package for Stochastic Boosting. J Stat Softw 17(2):1–27
17. Eisen M (1999) Scanalyze `http://rana.lbl.gov/EisenSoftware.html`
18. Eisen MB, Brown PO (1999) DNA arrays for analysis of gene expression. Methods Enzymol 303:179–205
19. Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139
20. Friedman JH, Hastie T, Tibshirani R (2000) Additive logistic regression: A statistical view of boosting. Ann Stat 28:337–407
21. Giannakeas N, Fotiadis DI (2009) An automated method for gridding and clustering-based segmentation of cDNA microarray images. Comput Med Imaging Graph 33:40–49
22. Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, et al (1999) Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286(5439):531–537
23. Gonzalez R, Woods R (2002) Digital image processing, 2nd edn. Prentice Hall
24. Hastie T, Tibshirani R, Friedman J (2009) The Elements of Statistical Learning, Second Edition. Springer
25. Hautaniemi S, Edgren H, Vesanen P, Wolf M, Järvinen AK, Yli Harja O, et al (2003) A novel strategy for microarray quality control using bayesian networks. Bioinformatics 19(16):2031–2038
26. Blekas K, Galatsanos N P, Likas A, Lagaris I E (2005) Mixture model analysis of DNA microarray images. IEEE T Med Imaging 24(7):901–909
27. Lashkari DA, DeRisi JL, McCusker JH, Namath AF, Gentile C, Hwang SY, et al (1997) Yeast microarrays for genome wide parallel genetic and gene expression analysis. Proc Nat Acad Sci USA 94(24):13,057–13,062
28. Liu X, Zhang L, Li M, Zhang H, Wang D (2005) Boosting image classification with LDA-based feature combination for digital photograph management. Pattern Recognit 38(6):887–901
29. Peterson LE, Coleman MA (2009) Logistic ensembles of Random Spherical Linear Oracles for microarray classification. Int J Data Min Bioinform 3(4):382–297
30. Ruosaari S, Hollmen J (2002) Image analysis for detecting faulty spots from microarray images. In: Lange S, Satoh K, Smith C (eds) Proc. 5th Int. Conf. on Discovery Science (DS2002) (S. Lange, K. Satoh, and C. Smith, eds.), Springer, pp 259–266
31. Schapire RE (1990) The Strength of Weak Learnability. Mach Learn 5(2):197–227
32. Schena M, Shalon D, Davis RW, Brown PO (1995) Quantitative monitoring of gene expression patterns with a complementary cDNA microarray. Science 270:467–470
33. Smyth GK, Ritchie M, Thorne N, Wettenhall J (2005) Limma: linear models for microarray data. In: Bioinformatics and Computational Biology Solutions using R and Bioconductor, Springer, pp 397–420
34. Sturn A, Quackenbush J, Trajanoski Z (2002) Genesis: cluster analysis of microarray data. Bioinformatics 18(1):207–208
35. Valafar F (2002) Pattern recognition techniques in microarray data analysis: a survey. Ann N Y Acad Sci 980:41–64
36. Valiant LG (1984) A theory of the learnable. Commun ACM 27:1134–1142
37. Vapnik V (1995) The nature of statistical learningn theory. Springer-Verlag
38. Yang YH, Buckley MJ, Dudoit S, Speed TP (2002) Comparison of methods for image analysis on cdna microarray data. J Comput Graph Stat 11(1):108–136