Exhaustive comparison of colour texture features and classification methods to discriminate cells categories in histological images of fish ovary

E. González-Rufino and P. Carrión
Department of Computer Science, Campus Universitario As Lagoas
University of Vigo, Ourense (Spain).
E-mail: nrufino@uvigo.es and pcarrion@uvigo.es

E. Cernadas and M. Fernández-Delgado
CITIUS: Centro de Investigación en Tecnoloxías da Información da USC
Campus Vida, 15872, Santiago de Compostela, Spain.
E-mail: eva.cernadas@usc.es and manuel.fernandez.delgado@usc.es

R. Domínguez-Petit
Department of Fisheries Ecology, Instituto de Investigaciones Marinas-CSIC, Vigo (Spain).
E-mail: rosario@iim.csic.es

December 7, 2012

Abstract

The estimation of fecundity and reproductive cells (oocytes) development dynamic is essential for an accurate study of biology and population dynamics of fish species. This estimation can be developed using the stereometric method to analyse histological images of fish ovary. However, this method still requires specialized technicians and much time and effort to make routinary fecundity studies commonly used in fish stock assessment, because the available software does not allow an automatic analysis. The automatic fecundity estimation requires both the classification of cells depending on their stage of development and the measurement of their diameters, based on those cells that are cut through the nucleous within the histological slide. Human experts seem to use colour and texture properties of the image to classify cells, i.e. colour texture analysis from the computer vision point of view. In the current work, we provide an exhaustive statistical evaluation of a very wide variety of parallel and integrative texture analysis strategies, giving a total of 126 different feature vectors. Besides, a selection of 17 classifiers, representative of the currently available classification techniques, was used to classify the cells according to the presence/absence

of nucleous and their stage of development. The Support Vector Machine (SVM) achieves the best results for nucleous (99.0% of accuracy using colour Local Binary Patterns (LPB) feature vector, integrative strategy) and for stages of development (99.6% using First Order Statistics and grey level LPB, parallel strategy) with the species *Merluccius merluccius*, and similar accuracies for *Trisopterus luscus*. These results provide a high reliability for an automatic fecundity estimation from histological images of fish ovary.

Keywords: Histological image, fish ovary, fecundity, stereology, classification, colour texture analysis, pyramid decomposition, multiresolution analysis, fractal analysis, local binary patterns, wavelets, coocurrence matrix, sum and difference histogram, support vector machine, statistical classifiers, ensembles, neural networks.

1 Introduction

The description of the reproductive strategies and the assessment of fecundity are fundamental topics in the study of biology and population dynamics of fish species. Fecundity is also important as an indicator of stock production and a reference point for management and sustainable fisheries [1]. The importance of determining fecundity has led to many research efforts to provide easier, faster and low cost methods since many years ago [2]. Stereometry [3] is one of the most precise and accurate method to estimate fecundity from histological images, however it is very time-consuming and it needs specialized technicians, which difficults its use routinely. It is based on the stereological method, which relates tridimensional parameters of a structure (in our case the fish ovary) with bidimensional measures obtained from sections of the structure. This allows to estimate the number of ovary cells (called oocytes) belonging to each category from histological sections, which are routinely elaborated in the laboratory. Figure 1 shows some histological images of fish species *Merluccius merluccius* (in English, *European hake*). The fecundity estimation implies the measurement and classification of cells in the histological images. Specifically, the diameter of matured cells with nucleous must be measured, so that the cells must be classified according to the presence and absence of nucleous (classes *With Nucleous*, labeled as WN, and *Without Nucleous*, labeled as WTN, respectively). Besides, the areas of cells in the different stages of development must be calculated, which requires to classify the cells in three stages of development defined by the experts: *Cortical Alveoli* (labeled as AC), *Hydrated* (HID) and *Vitellogenic/Atretic* (V/AT). Unfortunately, the routinary fecundity estimation using stereometry is rarely developed, because it still requires much work and time of specialized technicians, even with the currently available software. In order to improve this support to technicians, we recently proposed a publically available software tool called Govocitos[1] which automatically estimates fecundity from histological images of fish ovary [4]. Our software includes the following modules: 1) unsupervised and supervised detection of matured cells in the histological image; 2) unsupervised classification of matured oocytes according to the presence/absence of nucleous and to its development stage: the experts can easily modify or supervise the unsupervised detection or classification of oocytes using the graphical interface; 3) automatic fecundity estimation using the matured oocytes which have been recognized and classified. The information required and calculated by Govocitos is supported by either local or web-based databases and XML

[1] https://forxa.mancomun.org/projects/govocitos

files, which allows to check the fecundity estimations in a later time. In this paper we only focus on the design and evaluation of the classification module, developing an exhaustive statistical evaluation which compares a wide variety of different colour texture techniques and classifiers for the two classification problems (nucleous and stages). The paper is organised as follows. Section 2 briefly overviews the related work on microscopic image analysis and colour texture analysis. Sections 3 and 4 describe the applied texture feature extractors and classifiers respectively. Section 5 discusses the experimental results, and section 6 summarizes the major outcomes.

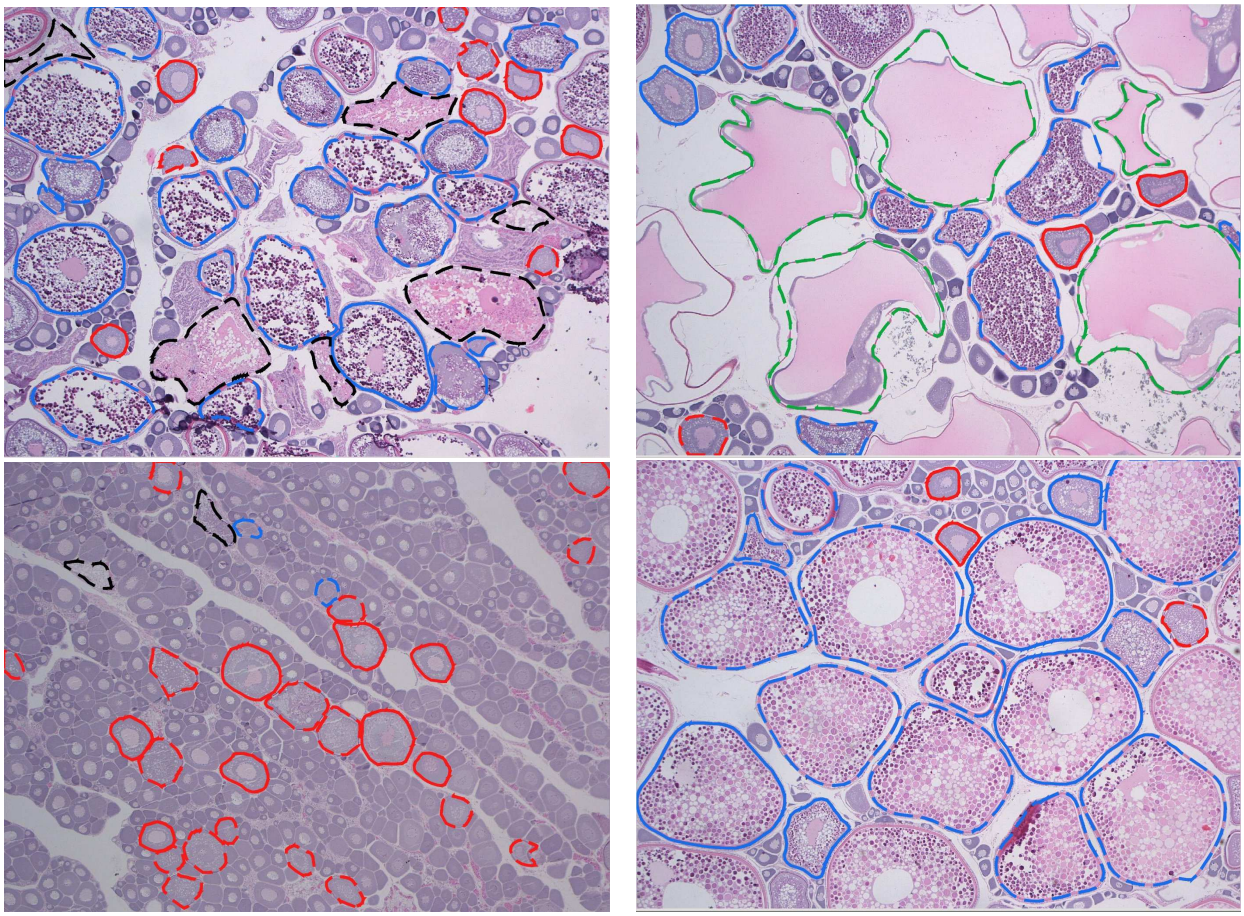

Figure 1: Examples of histological images of fish species *Merluccius merluccius*. The cell outlines were manually annotated by experts using Govocitos. The continuous (resp. dashed) line are cells with (resp. without) nucleous. The green, red, blue and black cell outlines represent respectively the *Hydrated*, *Cortical Alveoli*, *Vitellogenic* and *Atretic* stages of development.

2 Related work

The selection of an appropriate set of features is a fundamental challenge in pattern recognition problems [5]. In our problems of nucleous and stages classification, texture and colour seem to be relevant for the human expert perception. Nevertheless, while the ability of humans to distinguish different colour textures is clear, their automated description and recognition have been proven to be quite complex. Colour texture analysis relates the chromatic and textural properties of images. The approaches combining colour and texture can be grouped into parallel, sequential and integrative [6]. The **parallel aproach** joins the grey level texture features of the image to colour ones, and they are mainly used for image retrieval applications [7]. The **sequential approaches** use colour analysis to partition the image, followed by the processing of grey level texture of each region [8]. However, we already start from the image partitioned in regions (cells), so this approach is not interesting to our problem. The **integrative methods**, in their simplest version, use the union of the grey level texture features of each colour channel. The more sophisticated integrative methods imply the collective analysis of colour and texture properties. This analysis require vectorial computations, that are more complex and less intuitives than their scalar equivalents. Consequently, the majority of published works compute the texture features on grey level images [9, 10], or analyse grey level textures for each colour channel [11].

Grey level texture descriptors model the spatial relationship of a pixel and its neighbours, which provide information of the image structure (properties such as smoothness, coarseness and regularity). Experiments in colour texture analysis [9] conclude that the use of colour improves the performance of standard grey level texture analysis. However, most of the published works only use texture features [12, 13, 14, 15, 16]. Other examples of texture classification in computer vision applications are: fabric defect detection [12], quality evaluation and inspection of food [17, 18, 19], medical image analysis [20, 14] and remote sensing [21]. Texture features are also used to clasify objects in microscopical images [13, 22], and specifically in images of biological tissues (histological images) [14, 10]. However, in spite of the research done, we are no aware of any research that automatically analyses histological images of fish ovary.

In relation to colour space, Palm [6] achieved similar results comparing RGB and LUV colour spaces, using parallel and integrative approaches. Therefore, considering that most devices acquire RGB images, and that the colour conversion to LUV would require additional overhead, we choose the RBG colour space for our experiments. In the other hand, we combine texture features belonging to different families, following the recommendation of [23], which finds significant improvements in image segmentation when texture methods from multiple families are integrated. In the current work, we investigate the influence of the image properties (colour and texture), of the strategy to combine colour and texture features (integrative and parallel approaches), and of the classifier for nucleous and stages classification of cells in histological images of fish ovary. The cell shape is irregular, but many texture features like wavelet or Gabor transform must be applied on squared images with side power of two. In a preliminary work [24], we proved that the computation of texture features on irregular shapes instead on squared shapes significantly improves the accuracy of nucleous and stages classification. The following section describes the methods that we use in our experiments to extract colour and texture features,

adapted to be computed on irregular regions.

3 Feature extraction

Colour texture analysis can be tackled from different perspectives: simple colour features, grey level texture analysis, multiscalar (multiresolution or pyramid) grey level texture analysis and integrative colour texture analysis. Parallel approaches are directly derived as the union of colour and grey level texture analysis. The simplest integrative colour texture analysis is also derived as the union of the grey level texture features for each colour channel. This section is organized to briefly describe the most popular methods in each group adapted to operate on irregular regions. Let $G = \{0, \ldots, N_g - 1\}$, be the set of N_g quantized grey levels, S a finite subset of indexes specifying a region to be analysed (in our case, the cell of fish ovary), and $I(x, y) \in G$ the grey level in the pixel $(x, y) \in S$.

3.1 Statistical colour features

Let $B = \{r, g, b\}$ be the colour channels of a colour RGB image and let $I^p(x, y) \in G$ be the grey level in the pixel $(x, y) \in S$ of the channel $p \in B$. The histogram of an region S is the probability of a pixel $(x, y) \in S$ obtaining a certain value $i \in G$, and it is denoted as $P(i)$. It is normalized [25] dividing by the total number of pixels of region S. From the histogram for each channel $P^p(i), p \in B$, the following colour descriptors are derived:

- *CM (Colour Mean):* Mean value on each colour channel: CM $= \{\mu^r, \mu^g, \mu^b\}$ (3 features) and $\mu^p = \sum_{i=0}^{N_g-1} iP^p(i), p \in B$.
- *FOS (First-Order Statistics):* Provide information about the distribution of the grey levels that fall inside the region S. They are the simplest features to characterize images, with interesting properties such as invariance to geometric transformations. The vector FOS includes five statistics on each colour channel $p \in B$ (15 features in total): mean grey level (μ^p), variance (σ^p), third (m_3^p) and four (m_4^p) statistical moments, and entropy (H^p).

$$\mu^p = \sum_i iP^p(i) \tag{1}$$

$$\sigma^p = \sqrt{\sum_i (i - \mu^p)^2 P^p(i)} \tag{2}$$

$$m_q^p = \sum_i (i - \mu^p)^q P^p(i) \quad q \in \{3, 4\} \tag{3}$$

$$H^p = -\sum_i P^p(i) \log[P^p(i)] \tag{4}$$

where $i = 0, \ldots, N_g - 1$.

- *HFD (Histogram Fourier Descriptors):* The Fourier descriptors [26] of the histogram for each colour channel (9 features).

Since all these features are based on histograms, they are very sensitives to changes in illumination.

3.2 Grey level texture descriptors

These methods caracterize the local structure of a texture, being based on several local measurements like the relationship between the values of neighbouring pixels.

3.2.1 COMS (Co-Ocurrence Matrix Statistics)

The Grey Level Coocurrence Matrix (GLCM) of the image is based on the estimation of the second-order joint probability density function [27]. This matrix conveys information like the simultaneous occurrence of two values in a certain relative position. For the construction of a rotational invariant co-ocurrence matrix we consider all pairs of pixels that have a fixed distance (or scale) d from each other. Such $N_g \times N_g$-order matrices $\mathbf{C}(d)$ are only parametrised by d, and we have as many matrices as different distances or scales we choose. The element $C_{ij}(d)$ is the total number of pairs with values $i, j \in G$ at distance d from each other identified in the irregular region S:

$$C_{ij}(d) = \sum_{\mathbf{z}\in S} \sum_{|\mathbf{n}|=1} \delta[i - I(\mathbf{z})]\delta[j - I(\mathbf{z} + d\mathbf{n})] \tag{5}$$

where $\mathbf{z} \equiv (x, y)$, $\mathbf{n}$ is the unit vector pointing to a chosen direction, $I(\mathbf{z} + d\mathbf{n})$ is a grey value of another pixel that is at distance d from pixel $\mathbf{z}$ and the orientation defined by unit vector $\mathbf{n}$. In the above expresion $\delta(a-b) = 1$ if $a = b$ and $\delta(a-b) = 0$ otherwise. In practice, we create a list of pixels that are within a distance $d \pm 0.5$ from a given starting pixel. The number of neighbours for distances $d = 1, \ldots, 8$ are 8, 12, 18, 32, 28, 40, 40 and 48 respectively. When the image is scanned in a raster way, from top left to bottom right, only half of the pixels need to be visited. The probability density function $P_d(i, j)$ for each d is obtained dividing all elements of $\mathbf{C}(d)$ by the number of co-occurences for each d. This probability $P_d(i, j), i, j = 0, \ldots, N_g - 1$, is commomly characterized by the following features: energy, correlation, entropy, contrast and homogeneity:

$$\text{Energy} = \sum_i \sum_j P_d(i, j)^2 \tag{6}$$

$$\text{Correlation} = \frac{1}{\sigma_x \sigma_y} \sum_i \left[\sum_j i\, j\, P_d(i, j) - \mu_x \mu_y \right] \tag{7}$$

$$\text{Entropy} = -\sum_i \sum_j P_d(i, j) \log[P_d(i, j)] \tag{8}$$

$$\text{Contrast} = \sum_i \sum_j \frac{(i-j)^2 P_d(i, j)}{(N_g - 1)^2} \tag{9}$$

$$\text{Homogeneity} = \sum_i \sum_j \frac{P_d(i, j)}{1 + \mid i - j \mid} \tag{10}$$

where $\mu_x = \sum_i i \sum_j P_d(i,j)$, $\mu_y = \sum_j j \sum_i P_d(i,j)$, $\sigma_x = \sum_i (i-\mu_x)^2 \sum_j P_d(i,j)$ and $\sigma_y = \sum_j (j-\mu_y)^2 \sum_i P_d(i,j)$. All the sums over i,j are for $i,j = 0,\ldots,N_g-1$. The vector COMS includes the 5 features in eqs. 6-10 for $d=1$.

3.2.2 GLRLS (Grey Level Run Length Statistics)

A set of consecutive pixels in the image having the same grey level value is called grey level run [28], and the number of pixels is the run length. Once the run length matrix is calculated, the following features are derived: SRE (Short Run Emphasis), LRE (Long Run Emphasis), GLNU (Grey Level NonUniformity), RLN (Run Length Nonuniformity) and RP (Run Percentage). Tang [29] proposed also the following ones: LGRE (Low Grey Level Run Emphasis), HGLRE (High Grey Level Run Emphasis), SRLGE (Short Run Low Grey Level Emphasis), SRHGE (Short Run High Grey Level Emphasis), LRLGE (Long Run Low Grey Level Emphasis) and LRHGE (Long Run High Grey Level Emphasis). The run length matrix was also normalized by the number of runs in the region S. The vector GLRLS includes 11 features.

3.2.3 NGLDS (Neighbouring Grey Level Dependence Statistics)

This method considers the relationship between an element and all its neighbouring elements at one time. It is based on the calculation of a Neighbouring Grey Level Spatial Dependence Matrix (NGLDM) of the image [30]. The vector NGLDS includes 5 features derived from NGLDM: SNE (Small Number Emphasis), LNE (Large Number Emphasis), NNU (Number NonUniformity), SM (Second Moment) and ENT (Entropy).

3.2.4 SDH (Sum and Difference Histograms)

They were introduced by Unser [31] as an alternative to the usual co-occurrence matrices used for texture analysis with the advantage of decreasing the computation time and memory storage. Let $\mathbf{z}_k = (x_k, y_k) \in S$ and $\mathbf{z}_{k+d} \in S$ be two pixel elements separated by a distance d with grey levels $I(\mathbf{z}_k), I(\mathbf{z}_{k+d}) \in G$. For a relative displacement d, the sum and difference are define as $s_k = I(\mathbf{z}_k) + I(\mathbf{z}_{k+d})$ and $d_k = I(\mathbf{z}_k) - I(\mathbf{z}_{k+d})$ respectively. The normalized sum and difference histograms are defined as $P_s(i) = h_s(i)/N$ and $P_d(j) = h_d(j)/N$, where $h_s(i) = \mathrm{Card}\{\mathbf{z}_k \in S : s_k = i\}$, $h_d(j) = \mathrm{Card}\{\mathbf{z}_k \in S : s_d = j\}$, $N = \mathrm{Card}\{S\} = \sum_i h_s(i) = \sum_j h_d(j)$, where $i = 0,\ldots,2N_g-2$, and $j = -N_g+1,\ldots,N_g-1$. The resulting two vectors, $\mathbf{P}_s = \{P_s(0),\ldots,P_s(2N_g-2)\}$ and $\mathbf{P}_d = \{P_d(-N_g+1),\ldots,P_d(N_g-1)\}$ have dimension $2N_g-1$. Although these vectors can be directly used as texture descriptors, it may be necessary to reduce their dimensionality in order to use as an input for many classifiers. Statistical information can be extracted from the histograms by computing quantities as the mean, variance, statistical moments and entropy (equations 1-4) on both histograms $\mathbf{P}_s$ and $\mathbf{P}_d$. Unser also proposed a set of statistical features, which are equivalent to the most widely used features computed from the coocurrence matrices (see equations 6-10) as:

$$\text{Energy} = \sum_i P_s(i)^2 \sum_j P_d(j)^2 \tag{11}$$

$$\text{Correlation} = \frac{1}{2}\left[\sum_i (i-2\mu)^2 P_s(i) - \sum_j j^2 P_d(j)\right] \tag{12}$$

$$\text{Entropy} = -\sum_i P_s(i)\log[P_s(i)] - \sum_j P_d(j)\log[P_d(j)] \tag{13}$$

$$\text{Contrast} = \sum_j j^2 P_d(j) \tag{14}$$

$$\text{Homogeneity} = \sum_j \frac{P_d(j)}{1+j^2} \tag{15}$$

where $i = 0, \ldots, 2N_g - 2$; $j = -N_g + 1, \ldots, N_g - 1$, and $\mu = \sum_i iP_s(i)/2$. We test the performance of two feature vectors called SDH and SDHC. The former constains 10 features defined in eqs. 1-4 on histograms h_s and h_d. The SDHC vector encloses the 5 features defined in eqs. 11-15.

3.2.5 LBP (Local Binary Patterns)

Texture analysis using LBP was first proposed by Ojala et al. [32]. Local Binary Patterns extract local structures of images by comparing each pixel with its neighbours. At central pixel $\mathbf{z}_c = (x_c, y_c)$, each neighbouring pixel is assigned a binary label, which can be either 0 or 1, depending on whether the center pixel has higher intensity value than the neighbouring pixels (angularly evenly distributed sample points over a circle with radius R centered at the center pixel). The *rotation invariant* LBP label $LBP^{ri}_{R,P}(\mathbf{z}_c)$ for that center pixels $\mathbf{z}_c$, given the radius R and the number of involved neighbours P, is given by:

$$LBP^{ri}_{P,R}(\mathbf{z}_c) = \min_{0 \le n < P} \left\{ \sum_{i=0}^{P-1} u(\Delta I) 2^{(i+n)\%P} \right\} \tag{16}$$

where $\Delta I = I(\mathbf{z}_i) - I(\mathbf{z}_c)$, $\mathbf{z}_i$ is the i-th neighbouring pixel, $i = 0, \ldots, P-1$, $I(\mathbf{z}_i)$ is the grey level in pixel $\mathbf{z}_i$ and $u(x) = 1$ for $x \ge 0$, $u(x) = 0$ otherwise. The symbol % denotes the remainder operation. In our implementation, $R = 1$ and $P = 8$. Rotating an image causes the circular shifting effect of the binary labels at locations $\mathbf{z}_0, \ldots, \mathbf{z}_{P-1}$. This shift effect is removed by finding the minimum value among all the possible values of n in eq. 16. For $P = 8$, this minimum value only takes the following 36 rotation invariant patterns: LBP^{ri}={0, 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 37, 39, 43, 45, 47, 51, 53, 55, 59, 61, 63, 85, 87, 91, 95, 111, 119, 127}. We use as texture descriptors the following vectors: i) The normalized histogram of the LBP^{ri} patterns for pixels $\mathbf{z}_c \in S$, denoted by LBP^{ri}_h (36 features); and ii) From the original grey level image $I(x, y)$, we compute the image $I^{ri}(x, y)$ with the rotation invariant LBP, over which we calculate the grey level coocurrence matrix. The vector, called LBP^{ri}_c, includes the five texture features calculated in subsection 3.2.1 considering $I^{ri}(x, y)$ as input image.

Ojala [32] uses a subset of these patterns, called *uniform patterns*, LBP^{riu}. The uniform LBP refers to the patterns which have limited transitions from 0 to 1 or *vice versa*, lower or equal than two, in the circular binary representation. In [32], the uniform patterns account for around 90% of all patterns in a $R = 1$ and $P = 8$ neighbourhood, but [33] demostrated that this does not hold for all kind of images. Similarly to rotation invariant

patterns, we use as texture descriptors the normalized histogram of LBP^{riu} and the statistics calculated from the coocurrence matrix of LBP^{riu} image. The resulting texture vectors are called LBP_h^{riu} (10 features) and LBP_c^{riu} (5 features) respectively.

Given that in the LBP, rotation invariant or uniform, we simply calculate differences between the central pixel and its neighbourhood, LBP^{ri} or LBP^{riu} characterize the local image structure and they are not affected by any monotonic transformation of the grey scale. It is an excellent measure of the spatial pattern but, by definition, it discards contrast. Ojala et al. [32] propose to use the local variance as a measure of contrast:

$$v = \frac{1}{P}\sum_{i=0}^{P-1}[I(\mathbf{z}_i) - \mu]^2 \quad \mu = \frac{1}{P}\sum_{i=0}^{P-1} I(\mathbf{z}_i) \tag{17}$$

For each original grey level image, we calculate an image of real numbers containing the variance at each pixel. We define the vector LBP_s^v (5 features), which encloses the mean, variance, third and fourth statistical moment and entropy of the variance image. Besides, we define the image $I_v(x, y)$ as the variance image equalized to N_g grey levels. On the pixels $(x, y) \in S$ of $I_v(x, y)$, we calculate the statistics on the coocurrence matrix described in subsection 3.2.1. This texture vector is called LBP_c^v (5 features).

3.2.6 FA: Fractal Analysis

The fractal dimension is a useful metric for the analysis of images with self-similar content such a textures [34]. For images with limited resolutions and size, the fractal dimension D is related with the concept of self-similarity at some scales. Many methods exist to aproximate the fractal dimension of a grey-level or binary image, whose basis can be summarized in three steps: 1) Measure the quantities of the object using various step sizes; 2) Plot *log(measured quantities)* versus *log(step sizes)* and fit a least-squared regression line through the data points; and 3) Estimate D as the slope of the regression line. The most wide-spread literature methods can be grouped in three families: box-counting methods, fractional Brownian motion methods (which are not suitable for fractal analysis of irregular regions because they are based on Fourier spectrum analysis) and area measurement methods. Independently of the estimation method, the scale L over which we compute the estimation verifies $1 \leq L \leq \delta_{min}(S)$, where $\delta_{min}(S)$ is the minimum diameter of region S. Obviously, the number of scales to use influences the running time of the estimator. For all the fractal estimators, it is generally assumed that the pixel intensity $z = I(x, y)$ is considered as a point (x, y, z) in three dimensions, i.e. the image is a surface. The **box counting method** [35] counts the number of *boxes* which contain at least one pixel of the surface. Let us consider the image surface in a cube of size length$(S) \times$ width$(S) \times N_g$. This volume is divided into boxes with volume $L \times L \times L$. Let $N_b(L)$ be the number of boxes which contain at least one point lying on the surface. This value is counted for different box sizes L^k, where $k \in \{1, 2, \ldots, \lfloor \log_2(\delta_{min}(S)) - 1 \rfloor\}$. The fractal dimension D is estimated from the relation $N_b(L) \propto L^{-D}$. For the **probability method** [36], let $P(m, L)$ be the probability that there are m surface points within a box of size $L \times L \times L$ centered about an arbitrary point of the region surface. For any value of L we have $\sum_{m=1}^{N_p} P(m, L) = 1$, and $N_p = L^2$ is the number of possible points in the box. The number of boxes with size L needed to cover the whole region is computed as

$N_b(L) = \sum_{m=1}^{N_p} P(m, L)/m$. Again, D is estimated from $N_b(L) \propto L^{-D}$. The **blanket method** [37] estimates the area of the surface by first covering it with a blanket of thickness 2ϵ. This volume is then divided by 2ϵ to get an estimation of the area. The upper blanket $u_\epsilon(\mathbf{z})$ and lower blanket $b_\epsilon(\mathbf{z})$ are defined as:

$$u_\epsilon(\mathbf{z}) = \max\left\{u_{\epsilon-1}(\mathbf{z}) + 1, \max_{|\mathbf{m}-\mathbf{z}|\leq 1}\{u_{\epsilon-1}(\mathbf{m})\}\right\} \tag{18}$$

$$b_\epsilon(\mathbf{z}) = \min\left\{b_{\epsilon-1}(\mathbf{z}) - 1, \min_{|\mathbf{m}-\mathbf{z}|\leq 1}\{b_{\epsilon-1}(\mathbf{m})\}\right\} \tag{19}$$

for $\epsilon = 1, 2, \ldots$. Initially $u_0(\mathbf{z}) = b_0(\mathbf{z}) = I(\mathbf{z})$. The volume v_ϵ is computed as $v_\epsilon = \sum_{\mathbf{z}\in S}[u_\epsilon(\mathbf{z}) - b_\epsilon(\mathbf{z})]$. The area is computed as $A(\epsilon) = (v_\epsilon - v_{\epsilon-1})/2$. The fractal dimension D is estimated from $A(\epsilon) \propto \epsilon^{2-D}$. In either method the fractal texture is parametrised by only a scalar number, which it is not enough for a good characterization of the texture. Actually, most fractal based approaches describe the texture by only a few numbers, which are normally: 1) the fractal dimension of modified versions of the original image; or 2) the logarithmic texture parameters at multiple scales.

3.3 Multiscalar grey level texture features

One of the major developments in texture classification is the use of multiscalar features. Multiscalar analysis can be made by using neighbourhoods of different sizes (multiresolution) [32] or by using pyramid and multichannel transform algorithms such as Gabor filters or wavelet transforms [15]. Since Gabor filters use the Fast Fourier Transform (FFT) they are not suitable for irregular regions of interest. Combining pyramid decompositions with grey level texture features (section 3.2), we can also compute multiscalar features. In the following subsections, we describe the wavelet transform, and the multiscalar features.

3.3.1 Wavelet Transform

The dyadic wavelet transform is the most useful technique for multiscalar image analysis [38]. In practice, it is carried out using two channel filter banks composed of a low-pass (L) and a high-pass (H) filter and each filter bank is then sampled at a half rate (1/2 down sampling) of the previous frequency. By repeating this procedure, it is possible to obtain wavelet transforms of any order. With images, the wavelet transform is implemented in a separable way by filtering the rows and columns. The original image is transformed into four sub-images, namely low-low (LL), low-high (LH), high-low (HL) and high-high (HH). Wavelet transforms of any order are obtained repeating this process on the LL sub-image of previous order. Since our support region is irregular, we must adapt the general wavelet formulation to work with irregular regions. We extract the minimum power-of-two squared region that encloses S from the original image and we apply the wavelet transform on that squared region. The mean, variance and energy of the multiresolution transform coefficients for each subband at each decomposition level can be used for texture classification. Denoting by $I_{ij}(x, y)$ the j-th sub-image of the i-th level of decomposition, $i = 1, \ldots, N_l$, being N_l the number of decomposition levels and $j = \{LL, LH, HL, HH\}$, the resulting features are: $\mu_{ij} = \frac{1}{N_{S_i}} \sum_{(x,y)\in S_i} \mid I_{ij}(x, y) \mid$, $v_{ij} = \frac{1}{N_{S_i}} \sum_{(x,y)\in S_i} \mid$

$I_{ij}(x,y) - \mu_{ij} \mid^2$ and $E_{ij} = \frac{1}{N_{S_i}} \sum_{(x,y)\in S_i} \mid I_{ij}(x,y) \mid^2$, where N_{S_i} is the number of pixels of region S_i (i.e., region S at decomposition level i). Besides, μ_{ij}, σ_{ij} and E_{ij} are respectively the mean, variance and energy for the i-th decomposition level in the j-th sub-image. We define the feature vectors EWTAll, VWTAll and MWTAll as the energy, variance and mean respectively, computed on all sub-images (LL, LH, HL and HH) for each level of decomposition (vectors with 12 features for $N_l = 3$ levels of decomposition). We also define the feature vectors EWTLH, VWTLH and MWTLH as the energy, variance and mean respectively on the LL and HH sub-images for each level of decomposition (vectors with 6 features for 3 decomposition levels).

3.3.2 Multiresolution features

For the Co-Ocurrence Matrix Statistics (COMS) and Sum and Difference Histograms (SDHC) methods (see sections 3.2.1 and 3.2.4), the distance d between two points in the image describes the scale. The COMS and SDHC features are computed for different scales. The main problem is to choose the appropiate set of scales that effectively captures the structural information of the texture keeping a reasonable computational cost. These feature vectors are called MCOMS and MSDHC respectively (both with 4 scales, summing up 20 features).

For fractal analysis, we compute the logaritmic parameters (in the box counting method), the area parameters (in the blanket method) and the number of boxes (in the probability method) for different scales. Again, the main problem is to choose the appropiate set of scales. We carried out some experiments varying the scales to test its influence on the classification accuracy, selecting the following scales: 1) the box size $L = \{3, 5, 7, 9, 11, 13\}$ pixels for the probability method (vector MFDP, 6 features); 2) in the box counting method, the box size $L = \{2, 4, 8, 16, 32, 64, 128\}$ pixels (MFDBC, 7 features); 3) in the blanket method, $\epsilon = 1..20$ (vector MFDB with 20 features).

3.3.3 Pyramidal features

LBP can be extended to multiple scales in order to capture larger scale variations in texture patterns. Ojala et al. [32] propose the multiresolution LBP by the union of LBP descriptors using neighbours of different sizes (in particular, they use radius $R = 1, 2, 3$, and neighbourhood $P = 8, 16, 24$, respectively). Qian et al. [39] propose to represent local binary patterns in spatial pyramid domain (PLBP) archieving the bests results representing pyramid image by low-pass filters of wavelet transform. In order to apply PLBP to irregular regions, the lowest power-of-two squared region contaning the irregular region is extracted. We apply wavelet decomposition to this squared region and we calculate LBP on the down sampled irregular region of the LL sub-image for each level of decomposition. So, the texture feature vectors mentioned in section 3.2.5, LBP_h^{ri}, LBP_c^{ri}, LBP_h^{riu}, LBP_c^{riu}, LBP_s^v and LBP_c^v, are also applied on the Low-Low (LL) sub-image of the pyramid wavelet decomposition, resulting in the vectors $PLBP_h^{ri}$, $PLBP_c^{ri}$, $PLBP_h^{riu}$, $PLBP_c^{riu}$, $PLBP_s^v$ and $PLBP_c^v$. Finally, although we do not known any study about the use of pyramid decomposition with the grey level texture features COMS and SDHC (described in sections 3.2.1 and 3.2.4), we also define the texture vectors PCOMS and PSDHC, which are respectively COMS and SDHC features calculated on the down sampled irregular region in the LL

sub-image of the pyramid decomposition.

3.4 Integrative colour texture analysis

As mentioned, many colour texture analysis perform the analysis on each colour channel independently, and then join the texture features for each channel. But only a very few approaches study the relationship between colour channels altogether. Ivanovici and Richard [34] propose a colour version of the probability method to compute the fractal dimension of a grey level image. A colour image is an Euclidean hyper-space where each pixel is considered a 5-D vector (x, y, $r\,g$, b) in the RGB colour space. A direct extension of the fractal probability method (section 3.2.6) to colour images would count the pixels $F = f(x, y, r\,g, b)$ for which the Euclidean distance to the center $F_c = f(x_c, y_c, r_c\,g_c, b_c)$ of the hyper-cube is smaller than the box size L. Given that the Euclidean distance in the RGB space does not correspond to the perceptual distance between colours, the authors use the Minkowski infinity distance instead, defined by $|F - F_c| = \max_{i=1,\ldots,5}\{|f_i - f_{ci}|\} \leq L$. Given that the fractal dimension (one scalar) is not enough for a good texture characterization, we use a vector (called CMFP) with the number of boxes for the scales 3, 5, 7, 9, 11 and 13 pixels (6 features), as for the grey level version.

4 Classification

In the following subsections, we describe the different statistical, neural and ensemble classifiers which we use to classify cells in histological images of fish ovary. The colour texture feature vector of each cell is an input pattern, and it must be assigned to the class WN or WTN (for nucleous classification), and to the class AC, HID, or V/AT (for stages classification). In the following, n, N and M denote the number of inputs, training patterns and classes respectively. Besides, N_i, $\pi_i \equiv N_i/N$, $P(i|\mathbf{x})$ and $P(\mathbf{x}|i)$ denote the number of training patterns, *a priori* and *a posteriori* probabilities, and probability density function, of class i respectively.

4.1 Statistical classifiers

The **Linear Discriminant Analysis** (**LDA**) [40] is derived from the Bayes rule assuming that patterns belonging to class i, follow a normal (Gaussian) distribution with mean $\vec{\mu}_i$ and non-singular covariance matrix $\mathbf{\Sigma}$ common to all the classes. Under these hypotheses, the Bayes rule assigns a test pattern $\mathbf{x}$ to the class i with the highest posterior probability $P(i|\mathbf{x})$, given by:

$$-2log[P(i|\mathbf{x})] = (\mathbf{x} - \vec{\mu}_i)^T \mathbf{\Sigma}^{-1}(\mathbf{x} - \vec{\mu}_i) - 2log\pi_i + log|\mathbf{\Sigma}| \tag{20}$$

(where $|\mathbf{\Sigma}|$ is the determinant of $\mathbf{\Sigma}$), or, equivalently, to the class i which maximizes the linear function $L_i(\mathbf{x}) = 2\vec{\mu}_i^T \mathbf{\Sigma}^{-1}\mathbf{x} - \vec{\mu}_i^T \mathbf{\Sigma}^{-1}\vec{\mu}_i + 2log\pi_i$. The matrix $\mathbf{\Sigma}$ is approximated by the within-class covariance matrix $\mathbf{W} = (\mathbf{X} - \mathbf{GM})^T(\mathbf{X} - \mathbf{GM})/(n - M)$ where $\mathbf{X}$ is the $N \times n$-order training set matrix, $\mathbf{M}$ is the $M \times n$ matrix with the class means, and $\mathbf{G}$ is the $N \times M$-order matrix of class indicators ($G_{ij} = 1$ when the training pattern $\mathbf{x}_i$ belongs

to class j, and zero otherwise). We use the `lda` function (package `MASS`) in `R`, which employs a covariance $\mathbf{B}$ matrix weighted by the class prior probilities $\{\pi_i\}_{i=1}^{M}$. We tried different methods for the mean and variance estimations: MOMENT, MLE, MVE and T, achieving similar results.

The **Quadratic Discriminant Analysis** (**QDA**) [40] assigns a test pattern $\mathbf{x}$ to the class i which maximizes the quadratic function $Q_i(\mathbf{x}) = (\mathbf{x} - \vec{\mu}_i)^T \mathbf{\Sigma}_i^{-1} (\mathbf{x} - \vec{\mu}_i) + log|\mathbf{\Sigma}_i| - 2log\pi_i$. This function is the posterior probability $P(i|\mathbf{x}) = \pi_i P(\mathbf{x}|i)/P(\mathbf{x})$ by the Bayes rule, when each class i is normal multivariate with mean $\vec{\mu}_i$ and covariance matrix $\mathbf{\Sigma}_i$ (different for each class, as opposed to LDA). We use the `qda` function (package `MASS`) in `R`. In the stages classification QDA was not able to classify some feature vectors because some classes had few training patterns, leading to singular covariance matrices. The **Flexible Discriminant Analysis** (**FDA**) [41] estimates non-linear class boundaries using multivariate non-parametric regresion. We use the `fda` function (package `mda`) in `R`, which uses by default linear regression, but we also tried Multivariate Adaptive Regression Splines (MARS) [42] and Additive Spline Model by Adaptive Backfitting (BRUTO) [43], without improvements in the results. The **Mixture Discriminant Analysis** (**MDA**) [44] estimates the parameters of a FDA using the Expectation-Maximization (EM) method. We consider that each class i is composed by R_i subclasses $\{c_{ir}\}_{r=1}^{R_i}$. The subclass means $\vec{\mu}_{ir}$ are estimated initially using K-Means or LVQ, and the probability $P(c_{ir}|\mathbf{x}_j, i)$ of subclass c_{ir} is initialized as 1 if $\vec{\mu}_{ir}$ is the closest centroid to $\mathbf{x}_j$, and 0 otherwise. Afterwards, the EM method iteratively updates the means, probabilities and covariance matrix according to:

$$P(c_{ir}|\mathbf{x}_j, i) = \frac{P_{ir} e^{\frac{-D(\mathbf{x}_j, \vec{\mu}_{ir})}{2}}}{\sum_{k=1}^{R_i} P_{ik} e^{\frac{-D(\mathbf{x}_j, \vec{\mu}_{ik})}{2}}}, \quad \vec{\mu}_{ir} = \frac{\sum_{d_j=i} \mathbf{x}_j P(c_{ir}|\mathbf{x}_j, i)}{\sum_{d_j=i} P(c_{ir}|\mathbf{x}_j, i)} \tag{21}$$

$$\mathbf{\Sigma} = \frac{1}{N} \sum_{i=1}^{M} \sum_{d_j=i}^{N} \sum_{r=1}^{R_i} P(c_{ir}|\mathbf{x}_j, i)(\mathbf{x}_j - \vec{\mu}_{ir})(\mathbf{x}_j - \vec{\mu}_{ir})^T \tag{22}$$

where $D(\mathbf{x}_j, \vec{\mu}_{ir}) = (\mathbf{x}_j - \vec{\mu}_{ir})^T \mathbf{\Sigma}^{-1} (\mathbf{x}_j - \vec{\mu}_{ir})$ and $\mathbf{\Sigma}$ is the covariance matrix (common to all the classes). Besides, $P_{ir} \propto \sum_{d_j=i}^{N} P(c_{ir}|\mathbf{x}_j, i)$ (subclass probabilities) and $\sum_{r=1}^{R_i} P_{ir} = 1, i = 1, \ldots, M$. The updating finishes when $\mathbf{\Sigma}$ does not change any more. According to the Bayes rule, the MDA classifier assigns an input pattern $\mathbf{x}$ to the class i which maximizes $P(i|\mathbf{x}) = \pi_i P(\mathbf{x}|i) = \pi_i \sum_{r=1}^{R_i} P_{ir} e^{-D(\mathbf{x}_j, \vec{\mu}_{ir})/2}$. We use the `mda` function (package `mda`) in `R`, with default values for the number of subclasses ($R_i = 3, i = 1, \ldots, M$) and for the model dimension ($K = 5$). A **Generalized Linear Model** (**GLM**) [40] describes the classifier output y as a statistic distribution of the mean μ. We use the function `glm` (package `stats`) in `R`, with a *Binomial* model and logit link function for nucleous classification (because it is a two-class problem). The density is given by $\log[f(y)] = y \log\left(\frac{\mu}{1-\mu}\right) + \log(1-\mu) + \log\binom{1}{y}$, or equivally $f(y) = \binom{1}{y} \mu^y (1-\mu)^{1-y}$. For stages classification (three classes), we use a *Poisson* model with logarithmic link function, whose density is given by $\log[f(y)] = y \log \mu - \mu - log(y!)$. The **Multinomial Log-Linear Model** (**MLM**) is an extension of the binomial model for multinomial distributions (suited for both two- and multi-class problems). We use the `multinom` function (`nnet` package) in `R`, which uses a Multi-Layer Perceptron neural network, with the inputs scaled in the range $[0, 1]$. The **K-Nearest Neighbours** (**KNN**) classifier is applied using the `knn` function

(package `class`) in `R`, tuning the number k of neighbours with 25 values in the range $1 \leq k \leq 40$.

4.2 Neural Networks

The **Multi-Layer Perceptron** (**MLP**) is implemented using the `nnet` function (package `nnet`) in `R`, tuning the number H of hidden neurons in the range $3 \leq H \leq 15$ (6 values), and trying 10 random weight initializations. We use the C++ interface of LibSVM [46] to implement the **Support Vector Machine** (**SVM**) [45], with a Gaussian kernel whose spread γ tuned in the range $\{2^i\}_{i=-20}^{2}$ (23 values), and tuning the regularization parameter C in the range $\{2^i\}_{i=-5}^{14}$ (20 values). Other kernel types were not tried because the Gaussian kernel usually provides the best results. The **Radial Basis Function** (**RBF**) [47] neural network assigns a test pattern $\mathbf{x}$ to the class i which maximizes $\sum_{j=1}^{H} w_{ji} e^{-|\mathbf{x}-\mathbf{c}_i|^2/\gamma^2}$, where H is the number of hidden neurons and the weights w_{ij} are trainable parameters. We use the `newrb` function of the Matlab Neural Network Toolbox, which adds hidden neurons until sum-squared error reduces below a goal (0.1 in our case) or a maximum number (100) of hidden neurons is reached. We try 25 values for the Gaussian spread γ of the radial basis function in the range $1 \leq \gamma \leq 70$. The **Learning Vector Quantization** (**LVQ**) [48] is a nearest-neighbour method whose N_p prototypes or codebooks $\mathbf{m}_i, i = 1, \ldots, N_p$, are learnt using K-means and taking into account the desired class d_i for $\mathbf{x}_i$ and the class associated to each prototype. The LVQ output for a test pattern is the class associated to its nearest prototype. We use several LVQ implementations (functions `lvq1`, `olvq1`, `lvq2` and `lvq3`) included in the package `class` in `R`, achieving the best results with `lvq1`. We also tried the LVQ implementation of the Matlab Neural Network Toolbox, achieving much poorer results than the `R` version. The **Probabilistic Neural Network** (**PNN**) [49] assigns a test pattern $\mathbf{x}$ to the class i which maximizes $\sum_{d_j=i}^{N} e^{(\mathbf{x}_j^T \mathbf{x}-1)/\gamma^2}$, being N the training set size and γ the spread of the Gaussian neuron activation. We use the `newpnn` function included in the Matlab Neural Network Toolbox, tuning the values of γ in the range $0.01 \leq \gamma \leq 10$ (21 values).

The **Extreme Learning Machine** (**ELM**) [50] is a recent and promising single-layer feed-forward neural network whose output $\mathbf{y} \in \mathbb{R}^M$ for an input pattern $\mathbf{x} \in \mathbb{R}^n$ is defined as $\mathbf{y}(\mathbf{x}) = \sum_{i=1}^{H} \vec{\beta}_i f(\mathbf{w}_i^T \mathbf{x} + b_i)$, being H the number of hidden neurons, $\vec{\beta}_i \in \mathbb{R}^M$ the output weights, $f()$ the non-linear activation function (which must be infinitely differentiable), and $\{\mathbf{w}_i, b_i\}_{i=1}^{H}$ the weight vectors and biases of the H hidden neurons, randomly generated. The $H \times M$-order matrix $\mathbf{B} = [\vec{\beta}_1^T, \ldots, \vec{\beta}_H^T]^T$ is given by $\mathbf{B} = \mathbf{F}^\dagger \mathbf{G}$, where $\mathbf{G}$ is the class indicators matrix (see the LDA paragraph), the $N \times H$-order matrix $\mathbf{F}$ has elements $F_{ij} = f(\mathbf{w}_j^T \mathbf{x}_i + b_j), i = 1, \ldots, N; j = 1, \ldots, H$, and $\mathbf{F}^\dagger$ denotes the Moore-Penrose pseudo-inverse of $\mathbf{F}$. There is no training in ELM, because $\{\mathbf{w}_i, b_i\}$ are randomly generated and $\mathbf{B}$ is calculated analytically from the pseudo-inverse of $\mathbf{F}$. Given that the numbers H and M of hidden neurons and classes respectively is always low, the computational cost of this inversion is not high, so that the ELM is very fast. We use a Matlab implementation of ELM freely available[2], tuning the number H of hidden neurons in the range $18 \leq H \leq 200$ (16 values). We tried the following functions: hardlim $f(x) = 1$ for $x \geq 0$ and $f(x) = 0$ otherwise, sigmoid $(1 + e^{-x})^{-1}$, sinus $sin(x)$, triangular basis $f(x) = 1 + x$ for $x \in [-1, 0]$ and $f(x) = 1 - x$ for $x \in (0, 1]$, $f(x) = 0$ otherwise, and radial basis e^{-x^2}. The best results were

[2] http://www.ntu.edu.sg/home/egbhuang.

achieved using the sigmoid function.

4.3 Ensemble and decision tree classifiers

The **Recursive Partitioning** (**RPART**) classification tree [51] is a binary tree which partitions the input space in regions assigned to the available classes. We use the `rpart` function (package `rpart`) in `R` with the `class` splitting function, with parameters `minsplit` = 20 (minimum number of training patterns to split a node), complexity `cp` = 0.01 (minimum reduction in the training error to avoid discard a splitting) and `minbucket` = 7 (minimum number of patterns to avoid node removal). This function runs a 10-fold cross-validation in order to average and raise the reliability of the resulting tree. RPART is the base classifier for some of the following ensemble approaches. **Bagging** (**BG**) [52] reuses and selects training data to improve a classifier ensemble. Each base classifier is trained on a bootstrap sample of the training set, composed by $N' < N$ training patterns, some of which are repeated to achieve a training sample of N patterns (sampling with replacement). A voting is used to classify a test pattern. Bagging can improve the accuracy of unstable classifiers (e.g., RPART trees), which strongly varies with small changes in the data, achieving poor improvements with stable classifiers (e.g., KNN). We use the `bagging` function (package `ipred`) in `R`, with 100 bootstrap replications and the same parameters as RPART. A **Boosting** (**BT**) [53] ensemble is composed by an odd number of classifiers, with accuracy slightly better than the random classification, in order to achieve improvements with respect to a single classifier. Each classifier is trained using a bootstrap sample of the training set (without replacement) including patterns correctly and incorrectly classified by the previous classifiers in the ensemble. Each training sample must have similar size, and all the training patterns must be used at least for one sample. We use the `LogitBoost` [54] function (package `caTools`) in `R`, which uses a quasi-Newton method to fit an additive symmetric logistic model by maximum likelihood, using decision stumps (one node decision trees) as base classifiers, with 200 training iterations. **Adaboost** (**AB**) [55] is one the most popular versions of Boosting. The patterns appear in the training set according to their weight, which is updated during training, being larger for the difficult patterns. The classifier output is a voting among the ensemble classifiers, weighted according to its training error. We use the `boosting` function (package `adabag` in `R`), which implements Adaboost.M1 [56], with an ensemble of 100 trees. Using the Freund and Zhu [57] variants for the classifier weights we did not achieve better results than using the Breiman method (which is the default). Finally, **Random Forest** (**RF**) [58] learns an ensemble of classification trees, trained with different bootstrap samples of the training set. Each tree grows without pruning, selecting the best splits among m_{try} variables randomly selected, instead of choosing the best split among all the input variables as in RPART. The classifier output for a test pattern is selected by voting among the trees. The out-of-bag error (an estimation of the test error) is calculated aggregating the tree errors for the training patterns outside their bootstrap samples. We use the `rforest` function (package `RandomForest`) in `R`, with 500 trees and $m_{try} = n$ (the number of inputs).

Table 1: Notation used for feature vectors.

Method	#Feat.	Description
Statistical colour features		
CM	3	Mean value of each RGB channel
FOS	15	First-order statistics (eqs. 1-4) for each channel
HFD	9	Fourier descriptors of each colour-channel histogram
Grey texture descriptors with distance $d = 1$ and 8 neighbours		
COMS	5	Statistics from coocurrence matrix (eqs. 6-10)
GLRLS	11	Grey level run length statistics
NGLDS	5	Neighbouring grey level dependence statistics
SDH	10	First-order statistics of sum and difference histog.
SDHC	5	Features defined in eqs. 11-15
LBP_h^{ri}	36	Histogram of rotation invariant LBP
LBP_c^{ri}	5	COMS of rotation invariant LBP
LBP_h^{riu}	10	Uniform rotation invariant LBP
LBP_c^{riu}	5	COMS of uniform rotation invariant LBP
LBP_s^{v}	5	First-order statistics of local variance
LBP_c^{v}	5	COMS of local variance
Multiresolution grey texture methods		
MCOMS	20	Union of COMS features with distances $d = 1, 2, 4, 8$
MSDHC	20	Union of SDHC features with distances $d = 1, 2, 4, 8$
MFDBC	7	Multifractal Dimension: Box Counting with $L = 7$
MFDP	6	Multifractal Dimension: Probability with $L = 6$
MFDB	20	Multifractal Dimension: Blanket with $\epsilon = 20$
Pyramidal grey texture methods with $N_l = 3$ levels of decomp.		
XWTAll	12	Statistic X on each sub-image of wavelet decomp.
		X=E for energy,V for variance and M for mean
XWTLH	6	Equal to XWTAll but only for image LL and HH
PCOMS	20	Union of COMS features on the pyramid decomp.
PSDHC	20	Union of SDHC features on the pyramid decomp.
$PLBP_j^i$	$4x$	All pyramid LBP: x = #features of LBP_j^i
Integrative colour features		
CMFP	6	Colour Multi-Fractal Probability

LBP = Local Binary Patterns (section 3.2.5).

5 Results and discussion

In the first place, we describe the image collection and experimental settings (subsection 5.1). Afterwards, we compare the texture features described in section 3 for our data set. Given that there are 126 different feature vectors and 17 classifiers, we did not consider to apply every classifier on every vector. Instead, we developed a previous selection (subsection 5.2), classifying all the vectors with the same classifier, and discarding those vectors for which the classifier achieved very poor accuracy. We decided to use the Support Vector Machine (SVM), because it is considered a reference classifier for a variety of applications. The selected feature vectors are classified using the whole set of classifiers in order to find the best feature vector and classifier for nucleous and stages of development (subsection 5.3). Finally, further improvements to the experimental work are described in section 5.4.

5.1 Experimental settings

The fish ovaries in different maturity stages are embedded and sectioned with standard histological procedures. The sections are stained with *Haematoxylin–Eosin*, which produces a wide range of stained structures enhancing the image contrast. The images are obtained from the histological sections with a *LEICA DRE* research microscope connected to a *LEICA DFC320* digital camera using *LEICA IM50* software. The camera resolution

is 3.3 Mpixels (2088×1550 pixels), using squared pixels of 1.09 μm. The exposure time and colour balance are set automatically. In the evaluation, we use 47 histological images from 12 individuals of species *Merluccius merluccius*. For the experiments, the outline of cells were manually drawn and classified by human experts using the software Govocitos and saved in XML files. The available data set includes 1022 cells (aproximately 22 cells per image ranging from 9 to 47 cells). Their distribution per class is: 337 cells with nucleous (33% of the total) and 685 without nucleous (67%); for stages of development, 259 cells are in stage AC (25.3% of the total), 61 cells are HID (6.1%) and 702 cells (68.6%) are in V/AT. Note that a classifier which assigns all the patterns to the most populated class achieves 67% of accuracy for nucleous and 68.6% for stages. We built 10 groups of data sets using these patterns. Each group contains one training set, including 817 patterns (80% of the total), one validation set (102 patterns, 10%) and one test set (103 patterns, 10%), randomly selected. All of them were selected keeping the class and stage distribution: i.e., each training/validation/test set has about 33% and 67% for patterns with and without nucleous respectively, and analogously for stages (this constraint requires different training sets for nucleous and stages). We used a high percentage of training patterns (80%) in order to increase the significance of the training set. Each classifier was trained 10 times, once for each training test. For those classifiers which have some tunable pattern (e.g., the number H of hidden neurons on ELM and MLP, or the Gaussian spread γ for SVM, RBF and PNN), the selected value for each parameter was the one which maximized the average accuracy over the 10 validation sets. The validation sets were not used for those classifiers without tunable parameters. Each trained classifier was tested on its corresponding test set, reporting the average accuracy (in %), confusion matrix and class sensitivity and specificity over the 10 test sets.

5.2 Comparison among colour texture features

We compare the discrimination power of texture features described in section 3 on the data set described above using the SVM classifier. The colour texture classification on the original RGB histological images may be done using five strategies: 1) only colour descriptors; 2) only grey level texture descriptors; 3) the union of grey level texture and colour descriptors (parallel approach); 4) the union of the grey level texture descriptors for each colour channel (simple integrative approach); and 5) colour-texture descriptors (integrative approach). Table 1 shows these methods, their number of features and a brief description of the feature vectors used in the evaluation.

Table 2 shows the classification accuracy of the SVM using the texture vectors (see Grey texture descriptors in table 1) for nucleous and stages. The columns named "Grey level" are the grey level texture features, where only the intensity of the images is considered. In the columns named "Colour", the colour feature vectors are formed as the union of the grey level texture features for each colour RGB channel. In this case, the number of features is three times the number of features showed in table 1. Colour vectors achieve higher accuracies (for nucleous and stages) than grey level texture vectors (in average the difference is 3.7% for nucleous and 2.3% for stages). The best acuracies are achieved by the sum and difference of histograms method (vector SDH) using colour texture analysis (84.7% for nucleous and 93.4% for stages).

Table 2: Classification accuracy of SVM using grey level and colour texture vectors.

Method	Nucleous		Stages	
	Grey level	Colour	Grey level	Colour
GLRLS	**78.6**	78.6	88.9	90.6
NGLDS	74.3	79.7	89.9	91.5
COMS	75.2	83.2	**91.4**	92.3
SDH	76.5	**84.7**	90.5	**93.4**
SDHC	77.2	82.3	88.7	91.9
LBP_h^{ri}	74.8	71.6	90.0	90.2
LBP_c^{ri}	69.2	75.0	88.1	91.8
LBP_h^{riu}	76.5	77.7	90.8	90.3
LBP_c^{riu}	71.7	78.1	83.1	89.5
LBP_s^{v}	70.8	74.3	87.3	92.7
LBP_c^{v}	69.4	77.6	89.6	93.2
$\mu \pm \sigma$	74.5 ± 3	78.2 ± 4.3	89.6 ± 1.3	91.9 ± 1.2

Table 3: Classification accuracy of SVM with multiscalar methods using grey level and colour texture vectors.

Method	Nucleous		Stages	
	Grey-level	Colour	Grey-level	Colour
	Wavelet pyramid analysis			
EWTAll	71.1	71.0	89.3	90.7
EWTLH	71.7	74.0	89.4	90.9
MWTAll	68.5	72.8	80.4	83.4
MWTLH	68.3	72.8	78.6	84.6
VWTAll	73.0	78.8	87.1	89.9
VWTLH	73.0	79.2	87.2	90.0
	Multi-fractal analysis			
MFDBC	67.0	69.7	84.0	88.2
MFDP	74.8	79.4	89.1	91.7
MFDB	69.9	77.2	87.2	91.7
	Multiscalar coocurrence methods			
MCOMS	**80.3**	**85.1**	91.6	93.1
MSDHC	75.1	82.5	89.1	93.0
PCOMS	79.0	84.4	90.2	92.2
PSDHC	79.4	83.6	91.2	91.1
	Piramidal LBP			
$PLBP_h^{ri}$	73.2	77.1	91.9	90.6
$PLBP_c^{ri}$	75.5	79.2	**92.6**	92.4
$PLBP_h^{riu}$	79.2	82.1	92.6	93.0
$PLBP_c^{riu}$	76.4	80.5	90.5	92.0
$PLBP_s^{v}$	74.7	78.6	89.0	92.9
$PLBP_c^{v}$	80.2	83.6	89.6	**93.7**
$\mu \pm \sigma$	74.2 ± 4.2	78.5 ± 4.6	88.4 ± 3.8	90.8 ± 2.8

Multiscalar analysis can be made by using neighbourhoods of different sizes (multiresolution methods) or by using a unique neighbourhood size in a spatial pyramid image decomposition (table 1 lists both types of strategies). All pyramid images are generated by low-pass filters of wavelet transform using $N_l = 3$ levels of decomposition. Higher number of levels is not possible because some matured cells are quite small (minimum diameter is about 100 μm). Multiresolution methods are also applied to four scales (given by distances $d =$

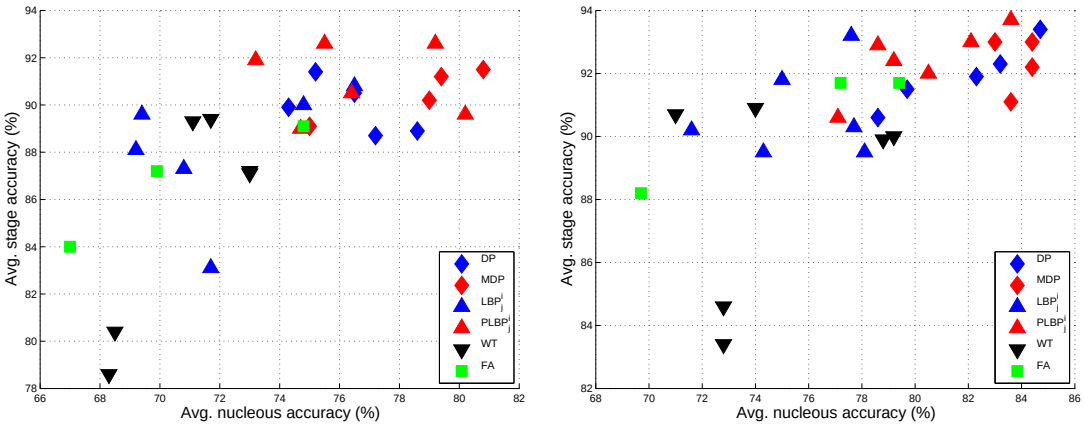

Figure 2: Classification accuracies in stages vs. in nucleous problems for grey level texture (left panel) and colour texture (right) from tables 2 and 3. The nomenclature used is: DP (vectors with spatial dependences among pixels: SDH, SDHC, COMS, GLRLS, NGLDS), MDP (vectors with multiscalar versions of DP: MSDHC, MCOMS, PSDHC and PCOMS), LBP (local binary patterns), PLBP (pyramid LBP), WT (wavelet transforms vectors) and FA (fractal features with vectors MFDBC, MFDP and MFDB). The use of colour texture features (right panel) raises the accuracy by 5% for nucleous and 1% for stages with respect to the left panel (grey level texture).

$1, 2, 4, 8$) selected in order to compare with pyramid approaches. Table 3 shows the accuracy of SVM using multiscalar texture vectors (see multiresolution and pyramid methods in table 1). As in table 2, the study is done on grey level and colour images for both problems. The tables 2 and 3 exhibit the same behaviour: the colour texture analysis achieves the best results for both classifications (85.1% for nucleous using colour MCOMS and 93.7% for stages using colour PLBP$^{v}_{c}$). The average improvement between colour and grey texture analysis is 4.3% for nucleous and 2.4% for stages. Comparing multiscalar and non-multiscalar texture vectors (tables 2 and 3), the results are similar on average. Nevertheless, all wavelet variants provide rather low accuracies, which are always superior when only the LL and HH sub-images are used instead of all sub-images. The reason may be that our textures do not exhibit any directionality. The best accuracy using wavelets is 79.2% for nucleous and the variance statistic (vector VWTLH), and 90.9% for stages using the energy (vector EWTLH). So, if we do not consider the wavelet and fractal families and we only calculate the difference between the multiscalar texture vector with its non-multiescalar equivalents (i.e. MCOMS or PCOMS with COMS), the improvement is significant only for nucleous classification (7.6% for grey level textures and 3.6% for colour texture in average), but not for stages. However, although the accuracy provided by wavelet vectors is not too high, they are computational fast and they will be combined with other texture families, following the suggestions of [23].

In order to report the best feature vector for both problems, figure 2 plots the accuracies for stages vs. nucleous, where texture vectors are grouped by families in: 1) DP: spatial dependences among pixels, which encloses the vectors SDH, SDHC, COMS, GLRLS, NGLDS; 2) MDP: multiscalar versions of DP; 3) LBP:

Table 4: Classification accuracies of SVM using combinations of grey level texture and colour descriptors (parallel approaches).

			A	B	C	D	E	F	G	H	I	J
			SDH	SDHC	COMS	MCOMS	PSDHC	LBP_h^{riu}	$PLBP_h^{riu}$	LBP_c^v	$PLBP_c^v$	MFDP
Keys	Vector	#Feat.	10	5	5	20	20	10	40	5	20	6
Nucleous classification												
1			76.5	77.2	75.2	80.3	79.4	76.5	79.2	69.4	80.2	74.8
2	FOS	15	81.8	83.9	83	85.1	84.6	84.4	82.6	84.5	83.7	84.2
3	FOS + EWTLH	21	83.6	86.6	85	83.8	85	85	81.8	85.8	82.9	85.7
4	FOS + MFDP	21	85.7	85.7	85.7	85.6	84.6	84	81.4	85.7	84.6	–
5	FOS + EWTLH + MFDP	27	86.6	86.3	**87.2**	85	84.4	84.4	82.1	85.5	84.2	–
Stages classification												
6			90.5	88.7	91.4	91.6	91.2	90.8	92.6	89.6	89.6	89.1
7	FOS	15	91.1	92.3	93.2	93.8	92.6	93.2	91.7	93.6	94	**94.8**
8	FOS + EWTLH	21	92.4	92.4	93.5	93.4	93.1	92.6	93	93.2	94.7	92.4
9	FOS + MFDP	21	93.5	92.2	92.7	93.6	93.2	93	92.5	93.6	94.1	–
10	FOS + EWTLH + MFDP	27	93.2	93.4	92	93.6	94	92.9	92.3	92.7	94.9	–

differents variants of local binary patterns; 4) PLBP: pyramid LBP; 5) WT: vectors based on wavelet transforms; and 6) FA: vectors based on fractal analysis. The left and right panels of figure 2 show respectively the grey level texture and colour texture classification (tables 2 and 3). The best features are in the upper right corner and the worst ones are closer to the lower left corner. In both panels, the WT and FA vectors are clearly the worst ones. Multiscalar techniques (MDP and PLBP) are the best features for grey level texture classification, but for colour texture analysis some DP vectors provide comparable performance. Comparing both plots, we can conclude that colour texture analysis (right plot) provides higher performances than grey level texture (left panel). The superiority is clear for nucleous classification, where the highest accuracy grows up from 80.3% for grey level vector MCOMS to 85.1% for colour MCOMS. In the stages classification, the accuracy grows up only from 92.6% for grey level LBP_c^{ri} to 93.7% for colour $PLBP_c^v$. This fact leads to conclude that the colour is an important property for all the texture features analysed in both classification problems, but specially for nucleous. On average, the performance improvement achieved comparing colour and grey level analysis are 4% and 2.4% for nucleous and stages respectively.

Although not included in any table, we also developed experiments using only colour feature vectors (section 3.1), with the following results:

- *CM*: achieves 71.1% and 85.5% for nucleous and stages respectively.
- *FOS*: 79.1% and 92.6% of accuracy for nucleous and stages respectively using RGB colour space. This colour space is not uniform, and it is more sensitive to illumination than uniform colour spaces like L*a*b*. We also used FOS features on chromatic coordinates a*b* (10 features), achieving 69.0% and 85.1% for nucleous and stages respectively. These worse results may be due to microscopy illumination is quite constant in the acquisition process.
- *HFD*: achieves 78.4% and 88.3% for nucleous and stages respectively. This vector includes three Fourier descriptors for each colour channel. Experiments with higher number of Fourier descriptors did not improve the results.

Table 5: Classification accuracies of SVM using grey texture and colour descriptors (parallel approaches).

No.	Combination of methods	#Feat.	Nucl.	Stag.
11	SDHC + LBP^v_c + FOS	25	85.2	93.5
12	MCOMS + LBP^v_c + FOS	40	85.9	92.2
13	SDHC + LBP^v_c + EWTLH + FOS	31	85.5	93.7
14	SDHC + LBP^v_c + MFDP + FOS	31	84.7	93.6
15	MCOMS + LBP^v_c + MFDP + FOS	46	**87.1**	94.1
16	SDHC + LBP^v_c + EWTLH + MFDP + FOS	37	85.9	92.6
17	SDHC + $PLBP^v_c$ + EWTLH + MFDP + FOS	52	84.6	**95.3**

Table 6: Classification accuracies of SVM using combinations of two colour texture methods (integrative approaches).

		K	L	M	N	O	P	Q	R	S	T	U	V
			SDH	SDHC	MCOMS	LBP^{riu}_h	$PLBP^{riu}_h$	LBP^v_c	$PLBP^v_c$	CMFP	MFDP	EWTLH	VWTLH
Key	Vector	#Feat. →	30	15	60	30	120	15	60	5	15	18	18
18			93.4	91.9	93.6	90.3	93	93.2	**93.7**	86.9	91.7	90.9	90
19	SDH	84.7	-	-	-	92.1	88.9	93.3	94	92.5	92.6	92.7	92.4
20	SDHC	82.3	-	-	-	90.6	93.4	93.5	91.7	93	92.9	92.5	92.8
21	MCOMS	**85.1**	-	-	-	91.3	93.8	93.2	93.5	93.5	93	93.6	93.2
22	LBP^{riu}_h	77.7	83.7	84.1	84.4	–	–	90.7	92.9	90.9	90.8	91.9	91.1
23	$PLBP^{riu}_h$	82.1	75.7	81.4	82.5	–	–	93.4	**94.3**	93.3	92.9	93	93.2
24	LBP^v_c	77.6	85.1	84.9	83.9	80.2	81.6	–	–	92.2	93.2	92.5	92.3
25	$PLBP^v_c$	83.6	84	83.5	84.5	81.9	83.4	–	–	93.8	94	92.2	91.4
26	CMFP	66	84.7	84	84.4	79.4	81.7	78.8	83.1	–	–	91.8	92.2
27	MFDP	79.4	85.5	**85.6**	84.9	80.9	81.6	81.6	82.7	–	–	92.5	92.4
28	EWTLH	74	82.6	84.6	83.5	78.6	81.9	81	82.6	74.7	81.9	–	–
29	VWTLH	79.2	82.7	82.3	84.2	82.8	82.4	82.6	82.9	81	81.7	–	–

Upper (resp. lower) triangular matrix for stages (resp. nucleous) respectively.

The performance achieved only by colour information is comparable to colour texture analysis for the stages classification and FOS vector (92.6% against 93.7%), but it is much lower for the nucleous classification (79.1% against 85.1%). This comparison confirms that colour texture analysis is necessary. Table 4 shows the accuracies of different parallel approaches for both problems. The rows 1 and 6 are the performance for differents grey texture vectors (columns A to J) for nucleous and stages respectively. The criteria used to choose the texture descriptors to be combined were a trade-off among: 1) its performance with both problems; 2) its computational complexity or number of features; and 3) whether the joint texture descriptors belong to different families [23], e.g., spatial dependences among pixels (SDH, SDHC and COMS features) and their multiscalar versions, local binary patterns (LBP^i_j and $PLBP^i_j$), fractal and wavelets. Due to the last criterion, we choose to combine vectors MFDP (fractal family) and EWTLH (wavelet family) despite of their low accuracies. The best performances in table 4 are 87.2% for nucleous and 94.8% for stages with vectors COMS+FOS+EWTLH+MFDP and MFDP+FOS respectively. The concatenation of colour information to grey texture always improves the results (compare rows 1 and 2 for nucleous, and rows 6 and 7 for stages in table 4). This improvement is higher for nucleous (7.9% in average) than for stages (3.2%). If we add the wavelet analysis (EWTLH vector, compare rows 3 and 2 for nucleous and rows 8 and 7 for stages), the average results improve in 1.3% for nucleous, but no improvement is observed for stages. It is important to emphasize the discrimination power of fractal features

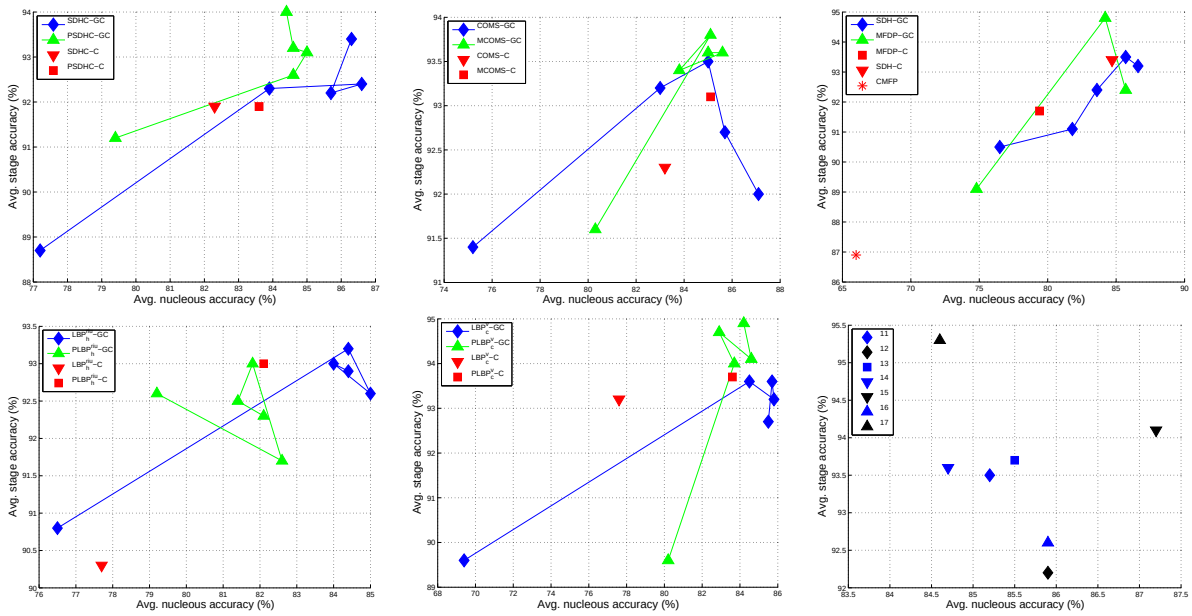

Figure 3: Classification accuracies of SVM in stages vs. in nucleous problems comparing parallel and integrative colour texture approaches. Red points are accuracies of colour texture features of table 6. Green (multiscalar methods) and blue lines are accuracies of the union of grey level texture and colour features in table 4. The first points of each lines are Grey Level Texture (GLT) classification, the seconds ones are GLT+FOS, the third ones are GLT+FOS+EWTLH, the fourth ones are GLT+FOS+MFDP and the last ones are GLT+FOS+EWTLH+MFDP. The lower right plot shows the results in table 5.

when they are combined with other techniques. For intance, the MFDP (fractal) + FOS vector achieves the highest score for stages problem (94.8%, line 7 of table 4) and a quite good score for nucleous problem (84.2%, line 2 in table 4) with only 21 features, much better than its grey level or colour texture accuracies (74.8% and 79.4% with grey and colour for nucleous, 89.1% and 91.7% with grey and colour for stages, in the line MFDP of table 3). Table 5 shows the accuracies of other combinations of texture and colour descriptors joining LBP and DP (dependences among pixels) families. The results do not improve for nucleous and they only increase in 0.5% for stages, reaching 95.3% with the vector SDHC + PLBP_c^v + EWTLH + MFDP + FOS, which is much more complex than MFDP + FOS.

Table 6 shows the performance of different colour texture descriptors (selected vectors of the columns named "colour" of tables 2 and 3) and the union of two vectors for both problems. The upper (resp. lower) triangle in the table 6 include results for stages (resp. nucleous). The column K and the row 18 are the accuracies for nucleous and stages using only one colour texture vector, and the rest of the table values are the accuracies for the union of two colour texture vectors. For stages classification, the best performance (94.3%) is achieved by the combination of PLBP_h^{riu} and PLBP_c^v, only 0.5% better than PLBP_c^v (93.7% in line 18 of table 6). For nucleous classification, the best score is 85.6%, only 0.5% better than MCOMS (85.1%, column K). We tested

some combinations of three vectors, but they did not improve the performance. Both for nucleous and stages, the performance achieved using these integrative approaches is lower than the parallel aproaches, which achieve 87.2% for nucleous and 95.3% for stages.

In order to compare the integrative and parallel approaches, figure 3 shows the classification accuracies in stages vs. in nucleous for differents feature vectors. The green, blue and black symbols represent feature vectors in the parallel approach, which are the union of one or various grey level texture vectors and colour features, and the red symbols are colour feature vectors (see table 6). For the left and middle plots, the green symbols are the multiscalar version of the method (the non-multiscalar version is represented by blue symbols). Excepting for PLBP_h^{riu}, the second points of the green or blue lines are placed upper and righter than the first ones, which means that the accuracy of the grey level texture features increases when are joint with the colour information (FOS). We want to emphasize the behaviour of fractal analysis (vector MFDP), which increases the accuracies in 5.7% for stages (from 89.1% to 94.8%) and 9.4% for nucleous (from 74.8% to 84.2%). The union of wavelet features (EWTLH vector) to GLT+FOS (second and third points in the lines) improves the accuracies for the simple methods SDHC, COMS and SDH. For LBP vectors, the union of EWTLH increases the accuracy for nucleous, but not for stages. For multiscalar vectors, the union of EWTLH decreases the accuracy. The reason may be that EWTLH provides multiscalar information, which is already implicit in the multiresolution/pyramid methods. The union of fractal information (MFDP vector) to GLT+FOS (the seconds and fourths line points) is not relevant for the majority of vectors analysed. In general, the multiresolution/pyramid vectors (green line) increase the accuracies with respect to their non-multiscalar versions (blue line) for stages, but decrease the accuracies for nucleous. Comparing the acuracies for colour texture vectors (red points), the multiscalar vectors improve the results (compare the squares to triangles in the left and middle plots): the best performance is achieved by MCOMS with accuracies of 85.1% for nucleous and 93.6% for stages. Many vectors in the parallel strategy achieve similar performances: 85.1% for nucleous and 93.8% for stages for MCOMS + FOS; 85.6% and 93.6% for MCOMS + FOS + MFDP; 85.7% and 93.6% for LBP_c^v + FOS + MFDP; 85.3% and 93.6% for SDHC + LBP_c^v + FOS + MFDP; and 86.9% and 94.1% for MCOMS + LBP_c^v + FOS + MFDP. Almost all include the fractal analysis (vector MFDP). Overall, the highest accuracies are 87.2% for nucleous and vector SDHC + FOS + MFDP + EWTLH and 95.3% for stages and vector SDHC + PLBP_c^v + FOS + MFDP + EWTLH.

5.3 Comparison among classifiers

The results achieved by SVM in the previous subsections are compared with other popular classifiers briefly described in section 4. We chose for comparison 25 feature vectors from tables 4, 5 and 6, in which SVM achieved the highest accuracies with moderate complexity or number of features. Table 7 shows the results comparing the accuracy of 17 classifiers over these 25 feature vectors for nucleous and stages problems, including the average accuracy for each classifier and vector, and the Friedman test ranking [59] for each classifier (decreasing with its accuracy). The best rank, accuracy, average accuracy, classifier and vector for both problems are in bold. The first column identifies the texture features: vectors from 2B to 5C belong to table 4, vectors from 11 to 17

Table 7: Classification accuracy for nucleous and stages using different classifiers and vectors.

Vector	AB	BG	BT	ELM	FDA	GLM	KNN	LDA	LVQ	MDA	MLM	MLP	PNN	QDA	RBF	RF	**SVM**	Avg.
Classification of nucleous																		
2B	78.3	76.8	77.5	80.3	78.3	79.3	71.7	77.5	73.7	74.9	80.6	78.3	72.5	70.0	83.9	77.1	83.9	77.3
2C	78.0	77.0	77.0	80.8	79.1	80.2	71.7	78.4	72.4	78.0	82.4	80.0	72.8	71.0	82.0	78.0	83.0	77.8
2D	80.7	77.5	77.6	80.1	80.3	82.5	74.4	79.5	75.8	80.5	85.1	80.0	75.1	74.9	84.4	79.3	85.1	79.6
2F	79.3	77.5	78.1	80.0	75.3	75.8	75.0	75.6	74.7	75.4	77.1	75.0	73.6	73.1	82.0	78.7	84.4	77.1
2H	77.0	76.7	75.0	79.3	77.7	78.6	70.1	77.4	71.8	77.6	79.1	76.9	71.0	74.2	84.0	78.1	84.5	77.0
2J	80.3	78.6	79.0	80.7	81.7	75.9	72.8	76.2	74.3	75.1	77.9	78.2	73.0	73.1	84.5	78.3	84.2	77.9
3B	78.9	77.3	76.6	79.8	80.0	79.5	72.1	78.3	71.0	77.4	79.9	79.2	71.7	73.4	85.2	77.5	86.6	77.9
3D	79.5	77.9	78.4	81.3	80.5	83.1	73.0	79.8	73.8	80.8	85.0	80.4	73.4	75.5	84.7	79.5	83.8	79.4
3H	75.0	76.7	76.1	78.8	79.0	79.8	71.0	78.9	70.5	77.4	79.9	77.0	69.2	75.7	84.8	77.5	85.8	77.2
4B	78.3	76.9	79.7	81.1	78.4	78.9	73.3	78.1	73.6	76.6	81.1	80.0	73.6	74.3	84.7	78.6	85.7	78.4
4D	83.2	78.7	80.9	80.3	79.9	82.2	73.9	79.7	75.7	80.6	85.0	79.6	73.9	77.2	84.5	81.5	85.6	80.1
5B	79.9	76.1	77.7	80.7	79.5	79.8	71.4	79.2	71.9	78.8	81.8	80.1	72.4	74.4	84.5	78.0	86.3	78.4
5C	78.6	77.7	76.4	80.7	80.8	81.7	72.4	80.3	72.5	78.8	84.3	80.7	72.7	77.0	85.3	78.3	**87.2**	79.1
11	78.8	77.3	76.5	81.9	78.6	79.8	71.6	78.5	72.1	76.6	81.4	77.7	72.4	71.4	85.6	78.6	85.2	77.9
12	80.3	77.9	79.7	80.3	80.0	82.0	74.9	78.8	75.2	79.2	84.1	81.3	75.4	75.7	85.5	80.2	85.9	79.8
14	79.3	77.5	78.2	81.1	80.2	82.0	72.0	79.0	71.8	78.3	83.2	80.5	73.6	75.0	84.9	78.5	84.7	78.8
15	83.3	80.0	79.4	80.5	81.1	82.1	73.9	80.7	74.7	80.1	84.2	79.6	74.9	78.3	84.8	82.1	87.1	80.4
16	78.9	78.1	79.0	78.5	80.7	81.9	72.9	79.7	71.0	79.0	83.1	78.8	72.5	78.5	85.0	78.5	85.9	78.9
17	80.5	80.4	78.3	76.6	80.8	82.9	71.3	79.3	72.5	81.3	82.6	78.6	73.6	79.0	83.1	79.1	84.6	79.1
19K	76.8	76.8	75.2	76.6	79.8	80.9	72.3	79.7	71.2	77.2	82.8	76.1	73.6	74.4	83.0	78.4	84.7	77.6
20K	81.0	79.6	82.1	81.0	78.8	80.1	75.8	78.3	77.2	78.3	82.3	80.4	78.1	64.7	82.7	79.5	82.3	79.0
21K	81.2	80.7	81.3	80.2	83.2	84.0	77.0	79.7	77.8	82.5	85.5	81.9	77.0	69.1	85.4	81.9	85.1	80.8
25K	80.5	78.9	75.7	78.4	79.7	80.7	76.8	75.3	76.7	80.6	82.9	78.2	78.6	66.7	82.0	80.6	83.6	78.6
25P	84.7	83.5	78.2	77.9	78.5	77.1	76.0	73.6	72.4	78.9	79.9	78.9	76.0	72.3	81.7	84.7	83.4	78.7
27M	81.9	81.2	78.0	81.9	79.1	81.2	73.7	78.4	75.5	77.9	81.6	80.0	75.3	73.5	84.3	80.7	85.6	79.4
Avg.	79.8	78.3	78.1	80.0	79.6	80.5	73.2	78.4	73.6	78.5	82.1	79.1	73.8	73.7	84.1	79.3	**85.0**	–
Rank.	7.2	10.7	10.7	6.8	7.2	5.6	15.9	9.9	15.4	9.4	3.5	8.8	15.0	14.9	2.1	8.3	**1.5**	–
Classification of stages																		
2B	92.9	90.2	92.7	92.0	90.6	75.0	89.9	89.9	87.7	90.7	91.1	91.5	91.3	87.3	82.7	91.8	92.3	89.4
2C	93.0	90.3	93.2	91.9	91.7	74.0	90.5	91.7	87.3	91.5	91.6	90.7	91.4	88.3	81.4	91.7	93.2	89.6
2D	92.7	90.7	92.2	92.6	93.2	74.8	88.9	91.7	86.8	91.9	92.2	91.1	90.9	90.6	80.1	92.4	93.8	89.8
2F	93.7	91.3	92.9	92.0	91.9	74.6	89.6	91.5	88.3	91.2	91.7	90.4	91.6	88.2	82.5	93.0	93.2	89.9
2H	91.7	90.3	91.3	92.2	93.1	72.6	89.7	92.8	86.2	92.0	91.7	91.4	91.7	89.3	81.9	91.6	93.6	89.6
2J	92.3	91.1	92.3	93.2	91.1	74.4	89.2	91.1	87.9	91.9	92.0	90.1	91.1	88.7	81.7	92.3	94.8	89.7
3B	92.3	91.1	92.3	93.4	91.1	74.4	90.1	91.9	87.9	91.9	92.0	90.9	89.5	88.6	85.1	92.3	92.4	89.8
3D	91.5	90.8	92.3	92.9	93.6	75.7	88.7	92.5	87.4	92.9	92.4	91.1	91.3	90.7	80.4	92.0	93.4	90.0
3H	91.7	90.5	92.4	92.9	93.5	74.1	88.2	93.1	85.5	93.3	92.0	91.6	90.8	89.7	81.7	91.8	93.2	89.8
4B	92.3	90.2	92.8	91.7	91.2	76.8	88.7	91.5	86.5	91.6	91.7	90.8	90.9	89.0	80.4	92.5	92.2	89.5
4D	83.5	90.7	92.8	92.8	93.2	76.9	89.2	92.1	86.8	92.8	91.7	92.0	90.8	89.8	81.8	91.8	93.6	89.5
5B	91.8	90.7	92.4	92.8	91.9	76.4	88.8	91.3	85.9	91.6	92.4	90.3	90.5	88.8	80.9	92.2	93.4	89.5
5C	92.0	90.5	92.7	92.7	92.4	78.7	89.0	92.7	86.7	91.8	92.1	91.6	72.7	89.2	82.0	92.4	92.0	88.9
11	92.1	90.4	92.4	92.9	92.4	75.9	89.3	91.8	86.8	91.9	91.6	91.4	91.1	89.6	82.6	92.1	93.5	89.9
12	91.7	90.9	92.2	92.8	93.2	76.2	89.1	92.1	87.3	92.3	91.7	91.8	91.8	90.5	81.3	92.3	93.2	90.0
14	91.9	90.5	91.3	91.5	92.7	77.1	88.6	92.7	86.2	92.1	92.1	91.9	91.0	89.6	82.6	92.1	93.6	89.9
15	92.9	90.9	93.0	92.3	93.5	77.3	88.7	92.4	87.2	92.2	91.7	90.4	91.5	88.1	80.4	92.0	94.1	89.9
16	92.1	90.4	91.9	93.1	93.6	77.9	89.2	92.9	86.7	93.1	92.4	91.5	91.7	86.0	81.8	92.1	92.6	89.9
17	93.9	91.1	93.2	92.6	94.5	77.2	89.8	94.3	87.4	94.6	93.7	91.4	91.5	–	82.2	91.9	**95.3**	90.9
19K	92.6	90.4	93.7	92.3	92.3	75.7	89.7	92.2	87.7	90.9	91.7	91.5	90.5	87.7	82.1	92.4	93.2	89.8
20K	90.8	90.5	91.9	92.3	89.5	69.6	88.6	89.5	85.5	88.7	91.3	90.1	88.3	86.4	79.4	90.4	91.9	87.9
21K	90.6	89.3	91.4	91.8	93.3	77.5	88.9	91.5	85.5	92.9	93.4	91.0	90.8	–	81.5	91.1	93.6	89.6
25K	90.9	90.0	91.0	90.8	94.3	77.3	90.2	93.6	86.1	94.0	91.1	90.6	91.3	–	82.3	90.6	93.7	89.9
25P	93.6	91.8	93.9	90.5	94.4	80.3	91.2	94.2	89.0	94.3	87.4	91.0	76.0	–	80.7	92.4	94.3	89.7
27M	90.8	90.5	91.9	91.7	90.1	76.2	88.3	90.5	85.2	87.9	91.6	90.1	75.3	87.8	79.8	90.1	92.9	87.7
Avg.	91.8	90.6	92.3	92.4	92.5	75.5	89.3	92.0	86.8	92.1	92.0	91.1	91.0	88.8	81.7	91.9	**93.4**	–
Rank.	6.1	10.9	4.8	4.4	4.4	16.9	12.9	6.3	14.5	6.4	6.7	9.9	10.7	13.8	15.8	6.3	**2.1**	–

belong to table 5, and vectors from 19K to 27M belong to table 6.

Analyzing the **nucleous classification**, the SVM achieves the highest accuracy (87.2%) for the vector 5C (FOS + EWTLH + MFDP + COMS, 32 features, table 4). Besides, SVM also achieves the highest average accuracy (85.0%) and the lowest ranking (1.5), being the best for 18 of 25 vectors, and achieving accuracies not more than 2.6% below the best on the remaining 7 vectors. The RBF is the second best classifier (ranking 2.1, 84.1% of average accuracy, being the best in 4 vectors), and the MLM is the third (ranking 3.5, 82.1% accuracy, it is the best for 2 vectors). There is a group of four classifiers (GLM, ELM, Adaboost and FDA) with similar results (accuracy about 80% and rankings about 5-7). Random Forest (which achieves the best accuracy for vector 25P), MLP, MDA, LDA, Bagging and Boosting achieve intermediate results (accuracies about 78%, rankings about 8-10), while KNN, LVQ, PNN and QDA are the worst ones (about 73%, ranking about 14-15). Comparing feature vectors, 5C and 15 (MCOMS + LPB_c^v + MFDP + FOS, 46 features, table 5)

provide the best accuracies (87.2% and 87.1%, achieved by SVM), and there are important differences between vectors: e.g., the SVM ranges 5% between the lower (82.3%, for vector 20K) and the highest accuracy; besides, the average accuracy varies in the range 77.2%-80.9% (about 4%) among vectors.

With respect to **stages classification**, again the best accuracy is achieved by SVM (95.3%), in this case using the feature vector 17 (SDHC + $PLBP_c^v$ + EWTLH + MFDP + FOS, 52 features, table 5). In the average, SVM achieves 93.4% of accuracy (ranking 2.1). The following classifiers are ELM, FDA and Boosting (ranking between 4.4-4.8, accuracy about 92%), and Adaboost, LDA, MDA, MLM (about 92%, ranking 6). Another group of classifiers includes Bagging, MLP and PNN (about 91%, ranking 10). Finally, the worst classifiers (accuracy below 90%, ranking above 12) are KNN, QDA, LVQ, RBF (despite its good result for nucleous) and GLM. The average accuracy over classifiers for stages range from 87.7% to 90.9% (3.2%), which is narrower than for nucleous. The vector 5C (the best for nucleous) achieves 3% less (92% with SVM). Conversely, the vector 17 for nucleous achieves 84.6% (2.6% less than 5C). We can conclude that none feature vector is the best both for nucleous and stages, and that vectors both 5C and 17 are required in order to achieve a good accuracy.

In order to evaluate the **complexity of feature vectors** for nucleous and stages classification, we calculated some of the complexity measures proposed in [60] including: F1: inverse of the maximum Fisher discriminant ratio for each pair of classes; F2: minimum class overlap for each dimension, over all the classes; F3: inverse of the maximum feature efficiency (defined as the fraction of the patterns separable by a feature); N2: ratio between the average within- and between-class; N3: error of a 1-Nearest Neighbour classifier; N4: non-linearity of a 1-NN classifier (1-NN error for an artificial set where each pattern is a linear interpolation of two training patterns of the same class); T1: inverse of the average order of the pattern adherence; T2: ratio between the number of features and patterns. In order to compare nucleous and stages classification, we averaged these complexity measures over the 25 feature vectors for nucleous and stages: the measures for nucleous are clearly higher for F1 (21.7 vs 0.17), F2 (0.53 vs. 0.29), F3 (24.4 vs. 3.4)), N3 (0.29 vs. 0.11) and N4 (0.3 vs. 0.08), which confirms that nucleous classification is harder than stages classification. However, comparing complexity and classification accuracy (e.g., using the SVM), the data set with the lowest average complexity (25P both for nucleous and stages) does not provide the best accuracy (achieved by vectors 5C and 17 for nucleous and stages respectively). Besides, there are low correlations between the complexity measures and the SVM errors for the different feature vectors.

Comparing **nucleous and stages classification**, all the classifiers achieve worse results in the former problem, which is clearly harder than the latter. The fig. 4 plots the stage vs. nucleous average accuracy for each classifier, where the SVM is placed on the top right corner (the highest accuracy for both problems). However, the classifier which are slightly worse for nucleous (RBF) is much worse for stages, and conversely the classifier which is slightly worse than SVM for stages (MLM) is much worse for nucleous (82% vs. 85% with SVM). The remaining classifiers are clearly distributed in two groups. The first one includes ELM, FDA, Adaboost, Random Forest, MDA, LDA, MLP, Boosting and Bagging. They are much worse than SVM for nucleous: about 78-80%, 5% below SVM (85%), but only slightly worse for stages (90%-92% for stages, vs.

93.4% for SVM). The second group (middle left of the plot) includes PNN, KNN, QDA and LVQ, with bad accuracies for stages (86%-90%) and very bad for nucleous (bellow 74%). It is surprising that popular classifiers as MLP, LVQ, KNN, LDA and QDA worked very bad for nucleous compared to other less-known classifiers (MLM, ELM, FDA, Adaboost, Bagging and Boosting). The RBF and GLM achieved very bad results for stages (about 81% and 75%), and GLM also worked poorly for nucleous (about 80%).

Considering the **classifier nature** (fig. 4), among the **ensemble classifiers** Adaboost (79.8%) and Random Forest (79.3%) are better than Bagging (78.3%) and Boosting (78.1%) for nucleous. For stages Boosting (92.3%), Adaboost (91.8%) and Random Forest (91.9%) work similarly well and Bagging (90.6%) is the worst. We also tested the single RPART classification tree, achiving results clearly worse (73.8% and 88.6% average accuracy for nucleous and stages respectively) than the RPART ensembles, so that it was not included in the table 7. Excepting the SVM, the best **neural network** was the ELM, which was worse than RBF, MLP and Boosting for nucleous, but it was the second better for stages (alongside with FDA). The RBF was good for nucleous and very bad for stages. The MLP was worse than ELM for both problems, and finally PNN and LVQ were among the worst classifiers. Considering the **statistical classifiers**, MLM was the best, with good results for nucleous and stages, followed by FDA, which was good for stages (92.5%) and very bad for nucleous (79.6%). LDA and MDA achieved equal results, slightly worse than FDA for nucleous, despite of being MDA a mixture of FDA classifiers. Finally, KNN, QDA and GLM were the worsts, the two formers mainly for nucleous and the latter mainly for stages.

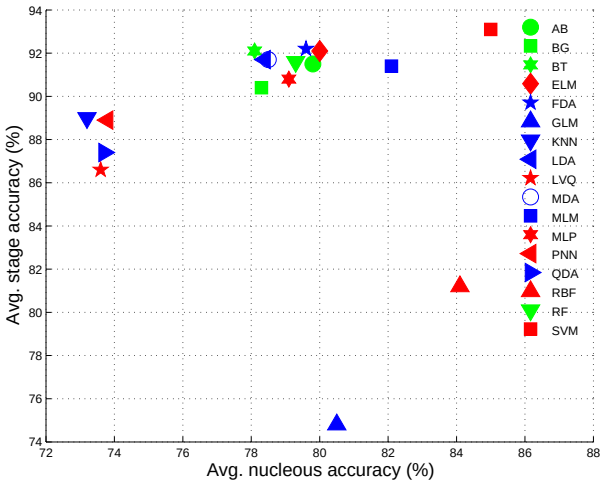

Figure 4: Average accuracies in stage vs. in nucleous classification for each classifier (statistics in blue, neural in red, ensembles in green).

5.4 Further experimentation

In order to find out the cause of the lower accuracies for nucleous classification, we develop some non-exhaustive experiments. Particulary, we detect that for some cells the classifier systematically fails, which suggests that

Table 8: SVM accuracy after the expert data revision.

		Nucleous		Stages	
Vector	#Feat.	CV	LOO	CV	LOO
2B	20	89.0	89.6	94.3	96.2
2C	20	89.5	89.0	95.0	96.5
2D	35	90.2	93.7	95.3	95.4
2F	25	89.8	92.6	93.8	**99.6**
2H	20	90.0	92	94.8	95.7
2J	21	90.5	91.0	**95.8**	97.7
3B	26	91.0	92.8	94.5	97.3
3D	41	90.7	92.0	95.1	96.2
3H	26	89.3	94.4	94.6	97.8
4B	26	91.5	92.1	94.4	96.5
4D	41	**92.0**	93.4	95.0	95.7
5B	32	91.0	94.2	94.5	97.4
5C	32	91.4	93.3	94.9	96.7
11	25	90.2	93.9	94.9	96.2
12	40	91.6	92.8	95.1	95.1
14	31	91.3	91.8	95.4	95.7
15	46	91.9	92.7	95.0	95.3
16	37	90.0	94.0	95.2	96.9
17	52	88.6	90.8	95.2	98.4
19K	30	88.0	94.5	94.5	96.3
20K	15	87.8	89.7	93.8	94.7
21K	60	91.0	91.2	94.5	94.9
25K	60	87.8	92.0	95	96.4
25P	180	85.8	**99.0**	95.5	96.4
27M	30	91.5	92.7	93.8	95.0
Avg.	–	90.0	92.6	94.8	96.5

the human experts made some mistakes annotating the cell class. To prove our hyphotesis, a more expertise technician checks the category of the whole data set and save in new XML files the changes. The category of 41 out of 1022 cells were changed, 31 for nucleous (2 from WN to WTN and 29 from WTN to WN) and 10 for stages (2 from V/AT to AC and 8 from AC to V/AT). These changes affect aproximately to the 1% and 3% of cells for stages and nucleous respectively. Additionally, given that the accuracy achieved in subsection 5.3 (classifier comparison) may be conditioned by the significance of the training set (related to the percentage of the patterns used for training, 80%), we also develop an experiment using the **Leave-One-Out** (LOO) validation, not conditioned because all the patterns except one are used for training in each trial. We run the SVM with the revised dates and the same feature vectors as in table 7, tuning its parameters C and γ using the same methodology as in the previous section (which will be called Cross Validation, CV). The results, and their averages over feature vectors, are reported in table 8, using CV and LOO. The CV column is devoted to see the change in accuracy due to expert revision, comparing with its corresponding value in table 7. The LOO column offers an accuracy measure which we think is more realistic than CV, limited by the training set significance. Regarding to **nucleous classification**, after the expert revision the best CV accuracy is 92.0% (vector 4D, 41 features), 5% higher than in table 7 (87.2%, vector 5C). This behavior is not limited to vector 4D, because the

average CV accuracy in table 8 is 90.0%, 5% above the value in table 7 (85.0%, column SVM of line "Avg"). This improvement (5%) is higher than the percentage of patterns (3%) modified by the expert revision: the reason may be that this revision removes contradictory information in the training set, which facilitates the learning and makes the SVM more reliable. With respect to LOO validation, the average accuracy is 2% higher than CV (92.6% vs. 90.0%): this means that, as suspected, CV is too pessimistic, probably due to the limited significance of the training set. Remarkably, the SVM achieves the "almost perfect" classification using LOO (99% of accuracy with vector 25P, 180 features). The fact that the whole data set is almost correctly classified using LOO, while CV achieves only 92% of accuracy, suggests that the random selection of the training set in Cross Validation is not adequate (despite of using 80% of the available patterns), and that the reduced test set (103 patterns) may not be representative enough to evaluate the classifier quality. Regarding to **stages clasification**, the best accuracies are 95.8% (vector 2J, 21 features) and 99.6% (vector 2F, 25 features) using CV and LOO respectively. Again, the SVM classifies almost correctly the whole data set using LOO, 4% more than CV, so the previous comments about CV and LOO are also valid. In this case, the expert revision only raises the average CV accuracy about 1% (94.8% in table 8 against 93.4% in table 7), similar to the percentage of patterns with stage revised by the experts. Therefore, both for nucleous and stages the SVM achieves 99% of accuracy with vectors 25P (PLBP_c^v + PLBP_h^{riu}, 180 features) and 2F (FOS + LBP_h^{riu}, 25 features) respectively, which can be considered a very reliable classification. The table 9 shows the confusion matrix and class sensitivities and specificities achieved by SVM with LOO. For nucleous (vector 25P), both classes WN (with nucleous) and WTN (without nucleous) exhibit sensitivities and specificities above 98%, with a small number of patterns missclassified (10 of 1022). For stages (vector 2F), the class HID is correctly classified (100% of sensitivity) despite of its low population, and only classes AC and V/AT are slightly mixed (4 missclassified against 252 and 705 patterns well classified) with sensitivies of 99.6%. In fact, given that the cell stage evolves in time from AC to V/AT, the domain experts agree that the border between the two stages is not clear, and that the discrimination criterion may change among experts. The Area Under Curve (AUC) values are also very high: 0.945 for nucleous and values above 0.986 for stages. The three pairs of stages were analized separately, because the LibSVM tool[3] used for ROC (Receiver Operating Characteristic) analysis only allows two-class problems.

Table 9: Accuracy (in %), confusion matrices, class sensitivity and specificity (in %), and Area Under Curve (AUC) achieved by SVM for nucleous and stages classification (species *Merluccius*, LOO validation).

Nucleous (vector 25P) Acc: 99.0%						Stages (vector 2F) Acc: 99.6%							
	WN	WTN	Se (%)	Sp (%)	AUC		AC	HID	V/AT	Se (%)	Sp (%)	AUC	
WN	360	5	98.2	98.6	0.945	AC	252	0	1	99.6	98.8	AC-HID	0.998
WTN	5	652	99.2	99.2		HID	0	61	0	100	100	AC-V/AT	0.986
						V/AT	3	0	705	99.6	99.8	HID-V/AT	0.998

The above results are refered to individuals of species *Merluccius merluccius*. In order to test the portability of these results to other species, we develop experiments with the species *Trisopterus Luscus* (a species of flatfish). Particularly, we use 31 histological images from 8 individuals of this species, summing up 912 cells

[3]http://www.csie.ntu.edu.tw/∼cjlin/libsvmtools/#roc_curve_for_binary_svm

(aproximately 30 cells per image ranging from 14 to 63 cells). Their distribution per class is: 385 cells with nucleous (42.2% of the total) and 527 without nucleous (57.8%); for stages of development, 525 cells are in stage AC (57.6% of the total), 14 cells are HID (1.5%) and 373 cells (40.9%) are in V/AT. Using the same methodology, we applied the SVM classifier to the 25 texture vectors of table 8 for the species *Trisopterus luscus*. The average results achieved are 91.6% for nucleous (resp. 96.3% for stages) using the CV, and 96.3% for nucleous (resp. 98.5% for stages) using LOO validation. The best LOO accuracies are achieved by vector 2F for nucleous (99.8%) and vectors 5B and 25P for stages (100%). The AUC values are also very high both for nucleous (0.914) and stages: 0.989 between AC and HID, 0.959 between AC and V/AT, and 0.941 between HID and V/AT. These results are even better than for species *Merluccius merluccius*, which confirms that this methodology reports certain grade of robustness with differents species, and that it is accurate enough to allow automatic classification for the species of interest typically considered.

6 Conclusions

The study of oocyte development dynamics and the automatic fecundity estimation in fisheries management requires the measurement of cells (subsequently the detection of nucleous), and the evaluation of the stage of development of cells in histological images of fish ovary. Both tasks lead to classification problems with two and three classes respectively (*With/Without Nucleous*, and *Cortical Alveoli, Hydrated, Vitellogenic/Atretic*). This analysis is performed over colour texture features extracted from the images. We developed an exhaustive comparison of colour texture features and classification methods in order to predict the presence/absence of nucleous and the stage of development. We tested a very wide variety of grey level and colour texture features belonging to different families: spatial dependences among pixels, Local Binary Patterns (LBP), wavelet texture analysis and texture fractal analysis. We compared colour texture features using parallel (the union of colour and grey level texture features) and integrative (the union of grey level texture over each colour channel) strategies, giving a total of 126 feature vectors. Additionally, the methods for calculating these features were modified to operate on irregular regions (cells) in the image. These vectors were used for both classification problems (nucleous and stages), covering the whole range of currently available classification techniques with a wide variety of approaches: statistical classifiers (LDA, QDA, FDA, MDA, GLM, MLM and KNN), neural networks (MLP, SVM, RBF, LVQ, PNN and ELM) and ensembles (Bagging, Boosting, Adaboost and Random Forest) of classification trees, up to 17 classifiers. For the species *Merluccius merluccius*, the SVM achieves the best results for almost all the feature vectors, reaching 87.2% and 93.4% of accuracy for nucleous and stages respectively using Cross Validation. However, further insights into the SVM results suggested us the presence of some labeling errors in the data set. An new expert revision confirmed these errors, changing the labels of about 3% and 1% of the patterns, and raising the SVM accuracies to 92.0% and 95.8%, for nucleous and stages of development respectively. Another factor what limits the SVM accuracy is the training set significance. In fact, SVM with Leave-One-Out validation achieves 99.0% and 99.6% of accuracy using vectors PLBP_c^v +

PLBP$_h^{riu}$ (colour LBP, 180 features, integrative method) and FOS + LBP$_h^{riu}$ (colour statistics and grey level LBP, 25 features, parallel method) for nucleous and stages respectively. These good results are also achieved with species *Trisopterus luscus* (99.8% for nucleous and 100% for stages), being accurate enough to allow an automatic fecundity estimation from histological images of fish ovary. Future work includes: 1) to develop further experiments with the other 13 species used in the daily work in the Institute of Marine Research-CSIC; 2) to evaluate the best colour texture features in a real enviroment using the software Govocitos; and 3) to experiment with uniform colour spaces.

Acknowledgment

This investigation was supported by the Xunta de Galicia (regional government) and Spanish Ministry of Science and Innovation (MICINN) under projects PGIDIT08MMA010402PR, TIN2012-32262 and TIN2009-07737 respectively.

References

[1] J. R. Hunter, J. Macewicz, N. C. H. LO, C. A. Kimbrell, Fecundity, spawning, and maturity of female Dover Sole, Microstomus pacificus, with an evaluation of assumptions and precision, Fishery Bulletin 90 (1992) 101–128.

[2] H. Murua, G. Kraus, F. Saborido-Rey, P. Witthames, A.Thorsen, S. Junquera, Procedures to estimate fecundity of marine fish species in relation to their reproductive strategy, J. of Northwest Atlantic Fishery Science 33 (2003) 33–54.

[3] L. S. Emerson, M. Greer-Walker, P. R. Witthames, A stereological method for estimating fish fecundity, J. Fish Biology 36 (1990) 721–730.

[4] R. Dominguez-Petit, J. M. P. Freire, S. Rábade, M. Fabeiro, A. D. Nieto, P. Carrión, N. Rufino, E. Cernadas, M. Fernández-Delgado, D. Dominguez-Vázquez, F. Saborido-Rey, A. Formella, New automatic software tool to estimate fish fecundity based on image analysis, in: Fish Reproduction and Fisheries, 2011.

[5] A. N. Jain, P. Duin, M. Jao, Statistical Pattern Recognition. A Review, IEEE Trans. Pattern Anal. Mach. Intell. 22 (1) (2000) 5–37.

[6] C. Palm, Color texture classification by integrative co-occurence matrices, Pattern Recognition 37 (2004) 965–976.

[7] S. M. Youssef, ICTEDCT-CBIR: Integrating curvelet transform with enhanced dominant colors extraction and texture analysis for efcient content-based image retrieval, Computers and Electrical Engineering in press (2012) in press.

[8] D. Zhang, M. M. Islam, G. Lu, A review on automatic image annotation techniques, Pattern Recognition 45 (2012) 346–362.

[9] A. Drimbarean, P. F. Whelan, Experiments in colour texture analysis, Pattern Recognition Letters 22 (2001) 1161–1167.

[10] A. Tabesh, M. Teverovskiy, H. Pang, V. P. Kumar, D. Verbel, A. Kotsianti, O. Saidi, Multifeature prostate cancer diagnosis and gleason grading of histological images, IEEE Trans. on Med. Imag. 26 (10) (2007) 1366 – 1378.

[11] J. Martínez-Alejaín, J. D. Luis-Delgado, L. M. Tomás-Balibrea, Automatic system for quality-based classification of marble textures, IEEE Trans. Syst., Man Cybern. C. Appl. Rev. 35 (2005) 488–497.

[12] H. Y. T. Ngan, G. K. H. Pang, N. H. C. Yung, Automated fabric defect detection - a review., Image and Vision Computing 29 (2011) 442–458.

[13] S. Kumar, S. H. Ong, S. Ranganath, F. T. Chew, Invariant texture classification for biomedical cell specimens via non-linear polar map, Computer Vision and Image Understanding 114 (2010) 44–53.

[14] O. S. Al-Kadi, Texture measures combination for improved meningioma classification of histopathological images, Pattern Recognition 43 (2010) 2043–2053.

[15] S. Li, J. Shawe-Taylor, Comparison and fusion of multiresolution features for texture classification, Pattern Recognition Letters 26 (2005) 633–638.

[16] T. Randen, J. H. Husoy, Filtering for texture classification: A comparative study, IEEE Trans. on Pattern Analysis and Machine Ingelligence 21 (4) (1999) 291–310.

[17] C. Zheng, D.-W. Sun, L. Zheng, Recent applications of image texture for evaluation of food qualities - a review, Trends in Food Science and Technology 17 (2006) 113–128.

[18] E. Cernadas, P. Carrión, P. G. Rodríguez, E. Muriel, T. Antequera, Analyzing magnetic resonance images of iberian pork loin to predict its sensorial characteristics, Computer Vision and Image Understanding 98 (2005) 345–361.

[19] P. Carrión, E. Cernadas, J. F. Gálvez, M. Damián, P. de Sá-Otero, Classification of honeybee pollen using a multiscale texture filtering scheme, Machine Vision and Applications 15 (2004) 186–193.

[20] I. Maglogiannis, C. N. Doukas, Overview of advanced computer vision systems for skin lesions characterization, IEEE Trans. on Inf. Technol. in Biomed. 13 (5) (2009) 721–733.

[21] W. Sun, G. Xu, P. Gong, S. Liang, Fractal analysis of remotely sensed images: A review of methods and applications, Int. J. of Remote Sensing 27 (22) (2006) 4963–4990.

[22] M. Rodríguez-Damián, E. Cernadas, A. Formella, M. Fernández-Delgado, P. D. Sa-Otero, Automatic detection and classification of grains of pollen based on shape and texture, IEEE Trans. Syst., Man, Cybern. C, Appl. Rev. 36 (4) (2006) 531–542.

[23] M. A. García, D. Puig, Supervised texture classification by integration of multiple texture methods and evaluation windows, Image and Vision Computing 25 (2007) 1091–1106.

[24] E. González-Rufino, P. Carrión, A. Formella, M. Fernández-Delgado, E. Cernadas, Statistical and Wavelet based texture features for fish oocytes classification, in: Lecture Notes on Computer Science, Vol. 6669, 2011, pp. 403–410.

[25] S. Theodoridis, K. Koutroumbas, Pattern recognition, Academic Press, 1999.

[26] L. da F. Costa, R. M. Cesar, Shape Analysis and Classification, CRC Press, 2001.

[27] M. Petrou, P. García-Sevilla, Image Processing: dealing with texture, Wiley, 2006.

[28] M. Sonka, V. Hlavac, R. Boyle, Image Processing, Analysis, and Machine Vision, Int. Thomsom Publishing (ITP), 1999.

[29] X. Tang, Texture information in run-length matrices, IEEE Trans. on Image Process. 7 (11) (1998) 16002 – 1609.

[30] L. H. Siew, R. M. Hodgson, and E. J. Wood, Texture measures for carpet wear assessment, IEEE Trans. Pattern Anal. Mach. Intell. 10 (1) (1988) 92 – 104.

[31] M. Unser, Sum and difference histograms for texture classification, IEEE Trans. Pattern Anal. Mach. Intell. 8 (1) (1986) 118–125.

[32] T. Ojala, M. Piatikäinen, T. Mäenpää, Multiresolution grey-scale and rotation invariant texture classification with local binary pattern, IEEE Trans. Pattern Anal. Mach. Intell. 24 (7) (2002) 971–987.

[33] S. Liao, M. W. K. Law, A. C. S. Chung, Dominant local binary pattern for texture classification, IEEE Trans. on Image Process. 18 (5) (2009) 1107–1118.

[34] M. Ivanovici, N. Richard, Fractal dimension of color fractal images, IEEE Trans. on Image Process. 20 (1) (2011) 227–235.

[35] N. Sarkar, B. B. Chaudhuri, An efficient box-counting approach to compute fractal dimension of image, IEEE Trans. on System, Man and Cybernetics 24 (1) (1994) 115 – 120.

[36] N. Sarkar, B. B. Chaudhuri, An efficient approach to estimate fractal dimension of textural images, Pattern Recognition 25 (9) (1992) 1035 – 1041.

[37] S. Peleg, J. Naor, R. Hartley, D. Avnir, Multiple resolution texture analysis and classification, IEEE Trans. Pattern Anal. Mach. Intell. 6 (4) (1984) 518 – 523.

[38] S. Mallat, A threary for multiresolution signal descomposition: the wavelet representation., IEEE Trans. Pattern Anal. Mach. Intell. 11 (7) (1989) 674–693.

[39] X. Qian, X.-S. Hua, P. Chen, L. Ke, PLBP: An effective local binary patterns texture descriptor with pyramid representation, Pattern Recognition 44 (2011) 2502–2515.

[40] W. Venables, B. Ripley, Modern applied statistics with S, Springer, 2010.

[41] R. T. T. Hastie, A. Buja, Flexible discriminant analysis by optimal scoring, J. of the Am. Stat. Soc. 89 (428) (1994) 1255–1270.

[42] J. Friedman, Multivariate adaptive regression splines, Annals of Statistics 19 (1) (1991) 1–141.

[43] T. Hastie, R. Tibshirani, Generalized Additive Models, Chapman and Hall, 1990.

[44] T. Hastie, R. Tibshirani, Discriminant analysis by Gaussian mixtures, J. of the Royal Stat. Soc. Series B (Methodological) 58 (1) (1996) 155–176.

[45] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[46] C. C. Chang, C. J. Lin, Libsvm: a library for support vector machines, `http://www.csie.ntu.edu.tw/~cjlin/libsvm` (2008).

[47] S. Haykin, Neural Networks and Learning Machines, Prentice Hall, 2008.

[48] T. Kohonen, Self-Organizing Maps, Springer, 1995.

[49] D. Specht, Probabilistic Neural Networks, Neural Networks 3 (1990) 109–118.

[50] Q. Z. G.B. Huang, C. Siew, Extreme Learning Machine: theory and applications, Neurocomputing 70 (2006) 489–501.

[51] C. S. L. Breiman, J. Friedman, R. Olsen, Classification and Regression Trees, Wadsworth, 1984.

[52] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.

[53] Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to Boosting, in: Proc. of the 2nd European Conf. on Computational Learning Theory, 1995, pp. 23–37.

[54] T. H. J. Friedman, R. Tibshirani, Additive logistic regression: a statistical view of Boosting, The Annals of Statistics 28 (2) (2000) 337–407.

[55] Y. Freund, R. Schapire, Experiments with a new Boosting algorithm, in: Proc. of the 13th Int. Conf. on Machine Learning, 1996, pp. 148–156.

[56] M. G.-M. E. Alfaro-Cortés, N. García-Rubio, Multiclass corporate failure prediction by Adaboost.M1, Int. Advances in Economic Research 13 (3) (2007) 301–312.

[57] S. R. J. Zhu, H. Zou, T. Hastie, Multi-class AdaBoost, Statistics and Its Interface 2 (2009) 349–360.

[58] L. Breiman, Random Forests, Machine Learning 45 (1) (2001) 5–32.

[59] D. Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, CRC Press, 2006.

[60] T. Ho, M. Basu, Complexity measures of supervised classification problems, IEEE Trans. Pattern Anal. Mach. Intell. 24 (3) (2002) 289–300.