Epithelial rosette detection in microscopic images

Kun Liu[1,3], Sandra Ernst[2,3], Virginie Lecaudey[2,3] and Olaf Ronneberger[1,3]

[1]Department of Computer Science [2]Department of Developmental Biology
[3]BIOSS Center for Biological Signaling Studies
University of Freiburg, Germany

This short article will explain how to detect a specific type of structures from (microscopic) images, focusing on a real task: the detection of epithelial cells organized radially, or rosettes, within a tissue. It is written for non-computer-scientists who are interested in using a computer program to automate or standardize some quantitative analysis related to object detection.

1 How does a trainable detector work

In Computer Vision techniques, the detection of any interested structures, patterns or constellations can be abstracted as *object detection*. Object detection has wide applications in many practical tasks, like face detection [1] and pedestrian detection [2]. It also has many successful applications in biomedical tasks. Approaches which are similar to our rosette detector are used for detecting cell mitosis [3] or specific structures in zebrafish embryos [4, 5].

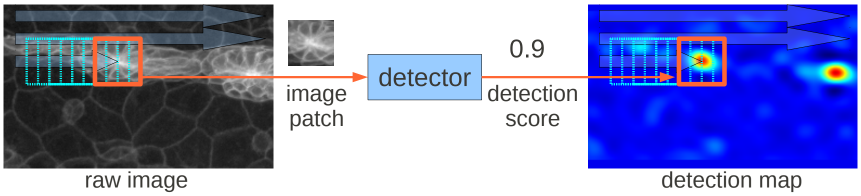

Fig. 1. An object detector goes through the whole image, checks every image patch, and produces a detection score map

As illustrated in Fig.1, an object detector can be considered as a *function* which maps every image patch (a small sub-region of image with predefined size) to an *detection score* value (ideally 0 or 1). This detection score indicates whether there is a target object at the center of the patch. To build such a detector, we always need some training data - images in which the rosettes have been manually identified and labeled, to help the computer program learn how to distinguish the target objects from other things (background or other objects). From the training data, we can have a lot of "image patch → desired detection score" pairs, based on which we can model and estimate the desired *function*.

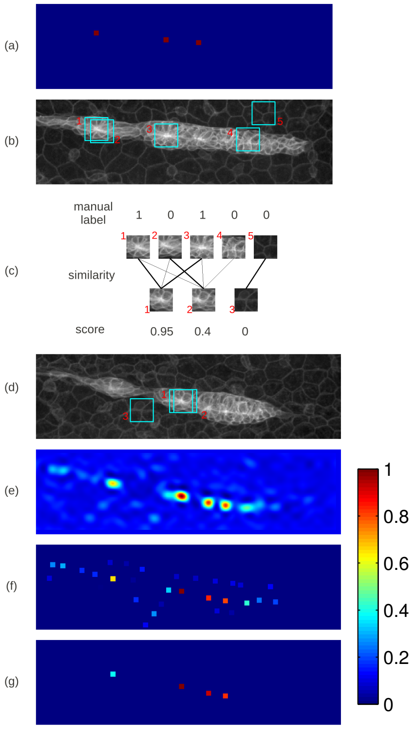

Fig. 2. How does a rosette detector work. **(a - b) Training data**: (a) manual labeling of the rosette positions as a binary image (b) microscopy image. **(c) Transfer the manual labeling to the test image based on the *similarity* comparison**: example patches are picked from image (b) and (d); similarities are illustrated by the thickness of the lines; the uncertainty in these relations leads to the real-value detection score. **(d - g) Test data and detection result**: (d) test image (e) detection map : response of the detector by applying the procedure in (c) on all positions). **(f - g) Post-processing**: (f) local maximum points (g) after a thresholding and a soft quantization.

One idea behind most common solutions is to find the image patches similar to the ones, which are manually labeled in the training data. Every image patch in the test image gets a score which results from its comparison with patches in the training images. Computing this score for all different patches, which can be densely sampled at each pixels, fulfills the detection job. See Fig.2a-e.

For the *similarity* comparison, there are definitely some uncertainties, *e.g.*, whether it is similar enough for propagating the labeling, or whether a patch p_1 is more similar to p_0 than p_2. So the estimation in general will not produce a binary (0/1) label for each position, but a real-value detection score between 0 and 1. It is actually very natural to get the real-value score in the output, because the image patches (and the underlying observed objects) have a continuous change in the appearance.

For more details about the basic object detection framework, refer to [6] as a good example.

2 More insight into the rosette detector

The technical way to implement a detection approach is not as straightforward as it is illustrated in Fig.2. We need to choose a *local image feature* to represent the important image property for each sub-patch; we need a *regional description* for representing all image content in a patch (which is usually too large for a *local feature* to describe); we need a statistics or machine learning approach to learn a function, which maps the feature/representation of an image patch to the detection score, and learn its parameters. In the following, we briefly explain the modules used in our rosette detector. We refer the readers to the references for more technical details.

Local image feature The local image content is described by densely sampled SIFT features (Fig.3) [7, 8]. This feature describes the orientation and coarse localization of image gradients (edge structures), which is exactly the information we need, to distinguish the rosettes from other image content. In our experiment, the SIFT feature consists of 4×4 subregions. Each subregion is about $2 \times 2 \mu m^2$ (which corresponds to 8×8 pixels with our microscopy).

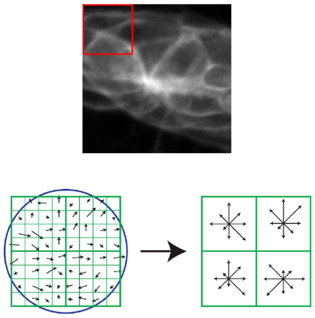

Fig. 3. Local image feature extraction. *Top*: an illustration of one sub-patch in an image patch. SIFT features are computed on sub-patches. A regional description for the whole image patch will be generated from these SIFT features. *Bottom*: the SIFT feature (from [7]): a descriptor is created by first computing the gradient magnitude and orientation at each pixel, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. These gradients are then accumulated into orientation histograms summarizing the contents over each subregion, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a descriptor using 2 × 2 subregions of 4 × 4 pixels.

Regional description A decision has to be made for each pixel, based on the image patch centered at that pixel. So all the image features computed on the sub-patches in a certain range should contribute to the decision. We choose to use a filter technique to "collect" all neighboring information [9]. A group of filters are computed efficiently on the features, which produce an output *vector* at each pixel, representing all image features in about $12.5 \mu m$ radius around the pixel (the radius of a rosette is normally about $10 - 15 \mu m$).

Learning From the training data, we can obtain many *vectors* from different patches, with their corresponding labels. Based on these, to estimate the mapping from a *vector* to a detection score can be as simple as a linear Least-Squares Fitting [9].

Locating objects From the above procedures, we have basically finished the training of the detector, which can produce a *detection map* as the one shown in Fig.2e. This map shows the detection score at each pixel. To get the exact positions of object, local maximum points are selected out (Fig.2f), because the detection score should be the highest at the object center[1]. Besides, the absolute magnitude of detection score has to be taken into account too, since a local maximum point with low detection score should not be recognized as a rosette. After applying a thresholding on the detection score and a re-scaling, the result is like Fig.2g.

3 Training and validation

In order to train the detector, we used 30 images of non-treated or DMSO-treated embryos with clearly distinguishable rosettes, and 17 images of embryos treated with the FGFR inhibitor SU5402 thus lacking rosettes. The positions of the rosettes were manually labeled using the "ImageJ" software.

A meaningful validation of the performance of the trained rosette detector can only be done experimentally. We used the standard cross-validation method to check the performance of the trained detector. We randomly choose half of the manually labeled data to train the detector, and use the other half to check if the detector produced the same result as the manual labeling, and repeat this for 5 times.

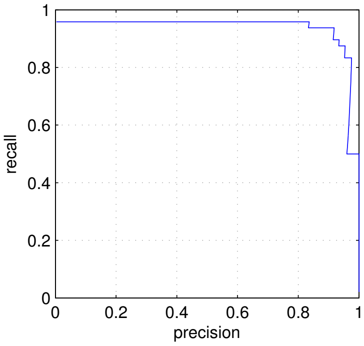

Fig. 4. Precision-recall curve of our rosette detector from cross-validation. Precision $= \frac{\text{Hits}}{\text{Hits + False Alarm}}$, Recall $= \frac{\text{Hits}}{\text{Hits + Misses}}$. The curve is plotted by varying the threshold.

There is always a precision/recall trade-off in detection tasks, *e.g.*, when we set a lower threshold for the step from Fig.2f to Fig.2g, we can get a higher recall with a lower precision (each true rosette gets a higher chance to be found by the detector, but meanwhile we also get more false-positives in the results). This ends up in the so-called precision-recall curve (shown in Fig.4). From the

[1] usually an additional non-maximum-suppression is carried out to remove the local maximum which is very close to but smaller than another local maximum. This is somehow not so critical in our approach, as the filtering technique guarantees a high smoothness in the detection map, which makes a object usually have only one corresponding local maximum

precision-recall curve, the detection performance is highly reliable. With a certain threshold (at the top right position of the curve) we can achieve 91.7% precision at 91.7% recall (this is the so-called Equal-Error-Rate point). This is a very good result, considering that the task itself is challenging even for a human expert.

Besides this quantitative test, the results are also checked against human perception qualitatively in more test images. Some example results are shown in Fig.6.

4 Counting the rosettes

From the extracted local maxima (Fig.2f), the standard method to determine the number of rosettes is to count only points with detection scores over a certain threshold. Due to the smooth transition from non-rosette to rosette structures, such a hard thresholding would introduce unnecessary quantization noise. As we are only interested in the relative number of rosettes for different biological experimental conditions, we can avoid this quantization noise and instead use the "confidence-weighted rosette number".

The "confidence-weighted rosette number" is computed as follows (as illustrated in Fig.2f-g): any candidate with a detection score lower than a preselected threshold t is discarded; the scores of the remaining candidates are renormalized into the range 0 to 1 (a detection score 1 will remain 1, but a detection score at the threshold will be shifted to zero); then the normalized values are summed up in each image separately to get the final "confidence-weighted rosette number" (*e.g.*, it is $0.95 + 0.89 + 0.83 + 0.38 = 3.05$ in Fig.2g.). This avoids some artifacts which could be caused by a hard thresholding, and also makes the threshold selection less critical to the statistics. See Fig.5 as an illustration of this effect.

The threshold t is then chosen to make the average "confidence-weighted rosette number" in the control group to be around 2.5, which is the expected average number of rosettes in the control group.

5 Applicability and limitation

The detector is trained from labeled samples, this leads to some advantages and limitations. The dependence on the training data is an intrinsic limitation of this method. It should be clarified that the trained detector can not "judge" the rosettes like a human. The actual job it does, is to find the objects (image patches) similar to the ones which are manually labeled in the training data, and the final confidence value somehow indicates the extent of the similarity. So if some imaging environment (like the microscopy setting) changes, it is possible that the trained detector no longer works for the new generated data. So in our experiments, we try to make the imaging condition as fixed as possible: all images were taken at the same microscope with the same objectives and laser intensities.

On the other side, we can always easily adapt the detector to some different imaging environment as long as sufficient manually labeled data is available in

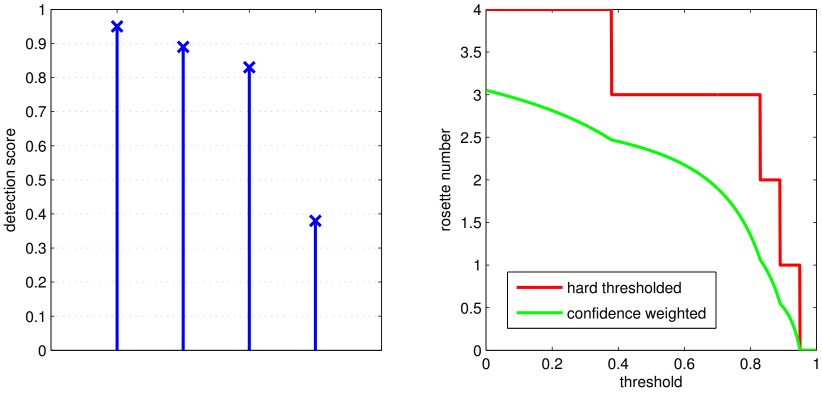

Fig. 5. The advantage of confidence weighting. Left: the detection score of the local maximal points used in this example. Right: how the threshold selection affects the output rosette number. The curves are plotted by varying the threshold from 0 to 1. It is clear that the hard thresholded result may "jump" when the threshold has a small change.

that environment. Another scenario is, when we observe the detector failed in some samples, we can manually label them and put them into the training set so that the detector can be improved. [2]

[2] The trained detector (in Matlab) is available at lmb.informatik.uni-freiburg.de/people/liu/rosette-detection. The full Matlab package including the detector training will also be available upon request.

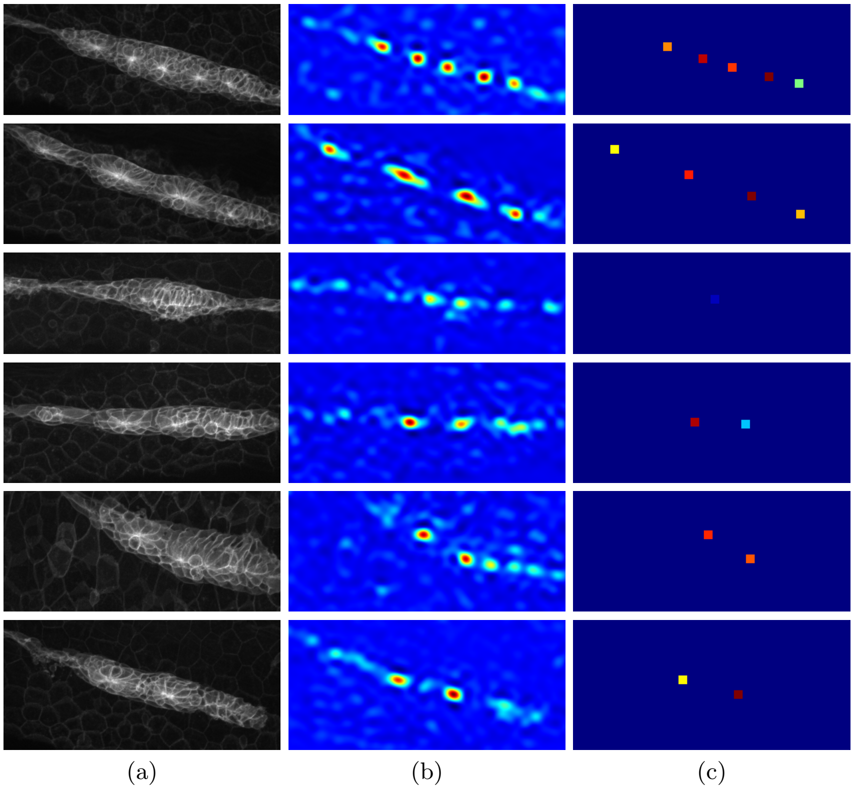

Fig. 6. Qualitative results of the rosette detector. (a) raw image. (b) output detection map. (c) detected rosettes with their confidence (after thresholding and re-scaling, shown by the same color coding as in Fig.2)

Bibliography

[1] P. Viola and M.J. Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005.

[3] M. Schlachter, M. Reisert, C. Herz, F. Schlurmann, S. Lassmann, M. Werner, H. Burkhardt, and O. Ronneberger. Harmonic filters for 3d multichannel data: Rotation invariant detection of mitoses in colorectal cancer. *IEEE Transactions on Medical Imaging*, 29(8):1485–1495, 2010.

[4] K. Liu, Q. Wang, W. Driever, and O. Ronneberger. 2D/3D Rotation-Invariant Detection using Equivariant Filters and Kernel Weighted Mapping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[5] O. Ronneberger, K. Liu, M. Rath, D. Rueß, T. Mueller, H. Skibbe, B. Drayer, T. Schmidt, A. Filippi, R. Nitschke, T. Brox, H. Burkhardt, and W. Driever. ViBE-Z: A Framework for 3D Virtual Colocalization Analysis in Zebrafish Larval Brains, 2012.

[6] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77(1):259–289, 2008.

[7] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[8] A. Bosch, A. Zisserman, and X. Muoz. Image classification using random forests and ferns. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–8, 2007.

[9] M. Reisert and H. Burkhardt. Equivariant holomorphic filters for contour denoising and rapid object detection. *IEEE Transactions on Image Processing*, 17(2):190–203, 2008.