FR-Ro-GE

Authors: Olaf Ronneberger, Robert Bensch, Philipp Fischer, Thomas Brox

Email: ronneber@informatik.uni-freiburg.de

Platform: Linux

Prerequisites: MATLAB 2014b (x64)

FR-Ro-GE: SUMMARY

Our approach for cell tracking is based on a deep convolutional neural network for segmentation and a greedy label propagation for tracking. The convolutional network takes the raw images as input and provides the final segmentation masks as output. The network architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. The network can be trained end-to-end from the few annotated images provided for the training datasets using extensive data augmentation with elastic deformations.

FR-Ro-GE: PREPROCESSING

In all datasets (except for **PhC-C2DH-U373**), the gray values are image-wise normalized (*normInt*) to unit range and zero median to compensate the changes in illumination during the recording. Two datasets (**DIC-C2DH-HeLa** and **Fluo-C2DL-MSC**) are downscaled (*scaleFactor*) to achieve a larger field of view and to compensate different recording resolutions.

FR-Ro-GE: SEGMENTATION

The segmentation is performed by a u-shaped deep convolutional network. It consists of a contracting path with a series of convolution, ReLU and max-pooling layers and an expansion path with a series of up-convolution, ReLU and convolution layers. In the expansion path, feature maps from the contracting path with the same resolution are copied (see [4] for the detailed architecture). The architecture of the network is an extension of the "fully convolutional network" [3]. One important modification in our architecture is that the upsampling part has a large number of feature maps, which allows the network to propagate context information to higher resolution layers. As a consequence, the expansive path is more or less symmetric to the contracting path, and yields the u-shaped architecture. The network does not have any fully connected layers and only uses the valid part of each convolution, i.e., the segmentation map only contains the pixels, for which the full context is available in the input image. This strategy allows the seamless segmentation of arbitrarily large images by an overlap-tile strategy. To

predict the pixels in the border region of the image, the missing context is extrapolated by mirroring the input image. The segmentation of a 512x512 pixel image takes less than a second on a standard laptop (equipped with a NVidia GTX 980m GPU). For the challenge contribution, we averaged the predicted segmentation maps of the input image and its mirrored versions.

Training the Segmentation Network. The loss function for training is computed by a pixel-wise soft-max over the final feature maps combined with a weighted cross entropy loss function [4]. We pre-compute the loss-weight map for each ground truth segmentation map to compensate the unbalanced class-frequency in the training data set, and to force the network to learn the small separation borders, that we introduce between touching cells. The loss-weight map is then computed as

$$w(x) = w_c(x) + w_0 \cdot \exp\left(-(d_1(x) + d_2(x))^2/(2\sigma^2)\right),$$

where w_c is the weight map to balance the class frequencies, $d_1{:}\Omega\rightarrow R$ denotes the distance to the border of the nearest cell and $d_2{:}\Omega\rightarrow R$ the distance to the border of the second nearest cell. In our experiments, we set $w_0 = 10$ and $\sigma \approx 5$ pixels. The provided manual segmentation masks in the training data set do not cover all visible cells. To obtain a consistent background training set, we manually created “ignore” -regions that cover all unlabeled cells, and set the loss-weights to zero within these regions. In deep networks with many convolutional layers and different paths through the network, a good initialization of the weights is extremely important. Otherwise, parts of the network might give excessive activations, while other parts never contribute. For a network with our architecture (alternating convolution and ReLU layers), a good initialization is achieved by drawing the initial weights from a Gaussian distribution with zero mean and a standard deviation of $\sqrt{2/N}$, where *N* denotes the number of incoming nodes of one neuron [1]. Data augmentation is essential to teach the network the desired invariance and robustness properties, when only few training samples are available. For microscopic images, we primarily need shift and rotation invariance and robustness to deformations and gray value variations. Especially, random elastic deformations of the training samples seem to be the key concept to train a segmentation network with a very low number of annotated images. We generate smooth deformations using random displacement vectors on a coarse 3 by 3 grid. The displacements are sampled from a Gaussian distribution with 10 pixels standard deviation. Per-pixel displacements are then computed using bicubic interpolation. Gray values of the input images are randomly scaled with a factor drawn from a Gaussian distribution with mean 1 and standard deviation 0.1. Drop-out layers at the end of the contracting path perform further implicit data augmentation. The augmented input images and their corresponding segmentation maps are used to train the network with the stochastic gradient descent

implementation of Caffe [2]. Due to the unpadded convolutions, the input image is larger than the output by a constant border width. To minimize the overhead and make maximum use of the GPU memory, we favor large input tiles over a large batch size and hence reduce the batch to a single image. To compensate for instable gradients, we accordingly set a high momentum (0.99) such that a large number of the previously seen training samples determine the current optimization step. We start the training with an initial learning rate of 0.001 which is decreased by a factor of 10 every 20,000 iterations. After 60,000 iterations (approx. 10 hours on an NVidia Titan GPU) the training is finished.

FR-Ro-GE: TRACKING

For tracking, we use a greedy algorithm. Each segment in frame t propagates its label to that segment in frame t + 1 with the highest overlap (measured as intersection over union). If a segment in frame t + 1 receives multiple labels, it prefers the segment in frame t with the highest overlap and discards the other labels. If a segment receives no label, a new label is assigned. For the dataset **PhC-C2DH-U373,** we applied two further processing steps that improved the results in our other submission. In each frame small segments below pixel area $a_{\min}$ are discarded. Additionally, the provided field of interest (FOI) specification is used to discard segments that lie completely outside the FOI (specified by the value *E*). However, we still use the tracking information from the full view to add parent links in case segments, which are tracked in the full view, reenter the FOI.

FR-Ro-GE: POST-PROCESSING

No post-processing is carried out after tracking.

REFERENCES

1. He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, 1026-1034 (2015).
2. Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R, Guadarrama S, Darrell T. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, (675-678) 2014.
3. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **39**, 640-651 (2017).

4. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of Medical Image Computing and Computer-Assisted Intervention*, 234-241 (2015).