MICROSCOPY CELL SEGMENTATION VIA CONVOLUTIONAL LSTM NETWORKS

Assaf Arbelle and Tammy Riklin Raviv

The Department of Electrical and Computer Engineering and The Zlotowski Center for Neuroscience
Ben-Gurion University of the Negev

ABSTRACT

Live cell microscopy sequences exhibit complex spatial structures and complicated temporal behaviour, making their analysis a challenging task. Considering cell segmentation problem, which plays a significant role in the analysis, the spatial properties of the data can be captured using Convolutional Neural Networks (CNNs). Recent approaches show promising segmentation results using convolutional encoder-decoders such as the U-Net. Nevertheless, these methods are limited by their inability to incorporate temporal information, that can facilitate segmentation of individual touching cells or of cells that are partially visible. In order to exploit cell dynamics we propose a novel segmentation approach which integrates Convolutional Long Short Term Memory (C-LSTM) with the U-Net. The network's unique architecture allows it to capture multi-scale, compact, spatio-temporal encoding in the C-LSTMs memory units. Promising results are presented.

1. METHODS

1.1. Network Architecture

The proposed network incorporates C- [1] blocks into the U-Net [2] architecture. This combination, as suggested here, is shown to be powerful. The U-Net architecture, built as an encoder-decoder with skip connections, enables to extract meaningful descriptors at multiple image scales. However, this alone does not account for the cell specific dynamics that can significantly support the segmentation. The introduction of C-LSTM blocks into the network allows considering past cell appearances at multiple scales by holding their compact representations in the C-LSTM memory units. We propose here the incorporation of the C-LSTM into the every scale in the encoder section of the U-Net. The network is fully convolutional and, therefore, can be used with any image size[1] during both training and testing. Figure 1 illustrates the full network architecture detailed in Section 2.

[1]In order to avoid artefacts it is preferable to use image sizes which are multiples of eight due to the three max-pooling layers.

1.2. Formulation

We address individual cells' segmentation from microscopy sequences. The main challenge in this type of problems is not only foreground-background classification but also the separation of adjacent cells. We adopt the weighted distance loss as suggested by [2]. The loss is designed to enhance individual cells' delineation by a partitioning of the image domain $\Omega \in \mathbb{R}^d$ into two classes: foreground, background, such that pixels which are near the boundaries of two adjacent cells are given higher importance. We set $\mathcal{C} = \{0, 1\}$ to denote these classes, respectively. Let $\{I_t\}_{t=1}^{T}$ be the input image sequence of length T, where $I_t : \Omega \rightarrow \mathbb{R}$ is a grayscale image. The network is composed of two sections of L blocks each, the encoder recurrent block $E_{\theta_l}^{\{l\}}(\cdot)$ and the decoder block $D_{\theta_l}^{\{l\}}(\cdot)$ where θ_l are the parameters. The input to the C-LSTM encoder layer $l \in [0, \ldots, L-1]$ at time $t \in T$ include the down-sampled output of the previous layer, the output of the current layer at the previous time-step and the C-LSTM memory cell. We denote these three inputs as $x_t^{\{l\}}$, $h_{t-1}^{\{l\}}$, $c_{t-1}^{\{l\}}$ respectively. Formally we define:

$$(h_t^{\{l\}}, c_t^{\{l\}}) = E_{\theta_l}^{\{l\}}(x_t^{\{l\}}, h_{t-1}^{\{l\}}, c_{t-1}^{\{l\}}) \tag{1}$$

where,

$$x_t^{\{l\}} = \begin{cases} I_t, & l = 0 \\ MaxPool(h_t^{\{l-1\}}), & 0 < l < L \end{cases} \tag{2}$$

The inputs to the decoder layers $l \in [L, \ldots, 2L-1]$ are the up-sampled [2] output of the previous layer and the output of the corresponding layer from the encoder denoted by $y_t^{\{l\}}$ and $h_t^{\{2L-1-l\}}$ respectively. We denote the decoder output as $z_t^{\{l\}}$. Formally,

$$y_t^{\{l\}} = \begin{cases} h_t^{\{l-1\}}, & l = L \\ UpSample(z_t^{\{l-1\}}), & L < l < 2L-1 \end{cases} \tag{3}$$

$$z_t^{\{l\}} = D_{\theta_l}(y_t^{\{l\}}, h_t^{\{2L-1-l\}}) \tag{4}$$

We define a network f_Θ with parameters Θ as the composition of L encoder blocks followed by L decoder blocks, and

[2]We use bi-linear interpolation

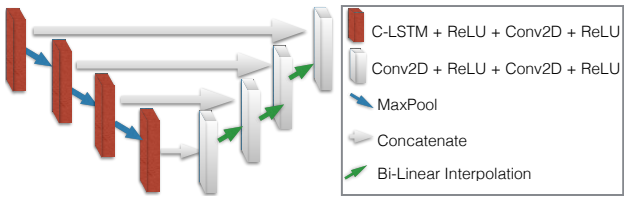

Fig. 1: The U-LSTM network architecture. The down-sampling path (left) consists of convolutional layers, ReLU activations, the output is then down-sampled and passed to the next layer. The up-sampling path (right) consists of a concatenation of the input from the lower layer with the parallel layer from the down-sampling path followed by a C-LSTM layer and a transpose-convlution up-sampling layer followed by ReLU activations.

denote $\Theta := \{\theta_l\}_{l=0}^{2L-1}$. Note that the encoder blocks, $E_{\theta_l}^{\{l\}}$, encode high-level **spatio-temporal** features at multiple scale and the decoder blocks, $D_{\theta_l}^{\{l\}}$, refines that information into a full scale segmentation map.

$$o_t \triangleq f_\Theta = z_t^{\{2L-1\}} \tag{5}$$

We set the final output as a $|\mathcal{C}|$-dimensional feature vector corresponding to each input pixel $\mathbf{v} \in \Omega$. We define the segmentation as the pixel label probabilities using the softmax equation:

$$p(c|o_t(\mathbf{v})) = \frac{\exp\{[o_t(\mathbf{v})]_c\}}{\sum_{c' \in \mathcal{C}} \exp\{[o_t(\mathbf{v})]_{c'}\}}, \quad c \in \mathcal{C} \tag{6}$$

The final segmentation is defined as follows:

$$\hat{\Gamma}_t = \arg_{c \in C} \max p(c|o_t(\mathbf{v})) \tag{7}$$

Each connected component of the foreground class is given a unique label and is regarded as an individual cell.

1.3. Training and Loss

During the training phase the network is presented with a full sequence and manual annotations $\{I_t, \Gamma_t\}_{t=1}^{T}$, where $\Gamma_t : \Omega \rightarrow [0, 1]$ are the ground truth (GT) labels. Given Γ_t, in order to ensure single cell separation, we define a weight map w_d that accounts for the Euclidean distances of the image pixels with respect to the nearest and the second nearest cells, d_1 and d_2 respectively.

$$w_d = w_0 \exp\left\{-(d_1 + d_2)^2 / {\sigma_0}^2\right\} \tag{8}$$

The network is trained using Truncated Back Propagation Through Time (TBPTT). At each back propagation step the network is unrolled to τ time-steps. The loss is defined using the weighted cross-entropy loss:

$$L = \sum_{t'=t}^{t+\tau} \sum_{\mathbf{v} \in \Omega} \sum_{c \in \mathcal{C}} (w_c + w_d)\delta(\Gamma_t \mathbf{v}, c) log(p(c)|o_{t'}(\mathbf{v}))) \tag{9}$$

where w_c is a constant for each class and δ is the Kronecker delta. The parameters Θ are updated using gradient descent.

2. IMPLEMENTATION DETAILS

2.1. Architecture

The network comprises $L = 4$ encoder and decoder blocks. Each block in the encoder section is composed of C-LSTM layer, leaky ReLU, convolutional layer, batch normalization [3], leaky ReLU and finally down-sampled using maxpool operation. The decoder blocks consist of a bi-linear interpolation, a concatenation with the parallel encoder block and an followed by two convolutional layer, batch normalization [3], and leaky ReLU. All convolutional layers use kernel size 3×3 with layer depths $(128, 256, 512, 1024)$. All maxpool layers use kernel size 2×2 without overlap. All C-LSTM and transpose convolutions kernels are of size 3×3 and 5×5 respectively with layer depths $(1024, 512, 256, 128)$. The last convolutional layer uses kernel size 1×1 with depth 2 followed by a softmax layer to produce the final probabilities (see Figure 1).

2.2. Training Regime

We trained the networks for approximately $100K$ iterations with an RMS-Prop optimizer [4] with learning rate of 0.0001. The unroll length parameter was set to $\tau = 5$ was set (Section 1.3) and the batch size was set to three sequences. The parameter for the weighted loss function were set to: $w_c = 1$, $w_0 = 10$ and $\sigma_0 = 5$ for all experiments.

2.3. Data

The images were annotated using two labels for the background and cell nucleus. In order to increase the variability, the data was randomly augmented spatially and temporally by: 1) random horizontal and vertical flip, 2) random 90^o rotation 3) random crop of size 160×160 4) random sequence reverse ($[T, T-1, \ldots, 2, 1]$), 5) random temporal down-sampling by a factor of $k \in [0, 4]$, 6) random affine and elastic transformations.We note that the gray-scale values are not augmented as they are biologically meaningful.

3. REFERENCES

[1] S. Xingjian, et al., "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Advances in neural information processing systems*, 2015, pp. 802–810.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, 2015.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448–456.

[4] G. Hinton, "Rms prop: Coursera lectures slides, lecture 6," .