Medical Image Analysis 55 (2019) 228–240

Contents lists available at ScienceDirect

Medical Image Analysis

journal homepage: www.elsevier.com/locate/media

Attentive neural cell instance segmentation

Jingru Yi[a,*], Pengxiang Wu[a], Menglin Jiang[a], Qiaoying Huang[a], Daniel J. Hoeppner[b], Dimitris N. Metaxas[a]

[a] Department of Computer Science, Rutgers University, Piscataway, NJ 08854, USA
[b] Lieber Institute for Brain Development, MD 21205, USA

ARTICLE INFO

Article history:
Received 18 October 2018
Revised 21 April 2019
Accepted 9 May 2019
Available online 10 May 2019

Keywords:
Neural cell
Instance segmentation
Cell detection
Cell segmentation

ABSTRACT

Neural cell instance segmentation, which aims at joint detection and segmentation of every neural cell in a microscopic image, is essential to many neuroscience applications. The challenge of this task involves cell adhesion, cell distortion, unclear cell contours, low-contrast cell protrusion structures, and background impurities. Consequently, current instance segmentation methods generally fall short of precision. In this paper, we propose an attentive instance segmentation method that accurately predicts the bounding box of each cell as well as its segmentation mask simultaneously. In particular, our method builds on a joint network that combines a single shot multi-box detector (SSD) and a U-net. Furthermore, we employ the attention mechanism in both detection and segmentation modules to focus the model on the useful features. The proposed method is validated on a dataset of neural cell microscopic images. Experimental results demonstrate that our approach can accurately detect and segment neural cell instances at a fast speed, comparing favorably with the state-of-the-art methods. Our code is released on GitHub. The link is https://github.com/yijingru/ANCIS-Pytorch.

1. Introduction

The cellular mechanism involved in the lineage path from a single neural stem cell remains mysterious in neuroscience. With the aid of real-time microscopy imaging system (Ravin et al., 2008), the specification of neurons, astrocytes, and oligodendrocytes from a neural stem cell could be recorded as a time-lapse video. Neural cell instance segmentation, which aims to detect and segment every cell in a microscopic image simultaneously, lays the foundation for many important neuroscience applications, such as exploring fate specification in neural stem cells. An accurate and fast instance segmentation tool can be applied to large microscopic video datasets and is thus crucial to the analysis of neural cell behavior. Nevertheless, there are many difficulties in neural cell instance segmentation. First, neural cells vary in shape and size. Second, many neural cells have tiny and slender structures, such as filopodia and lamellipodia, which are essential to neural cell behavior analysis but quite difficult to segment. Third, neural cells tend to adhere to each other due to cell interaction. Last but not least, neural cells may have obscure contours, and microscopic images usually have low contrast and background impurities. These challenges are illustrated in Fig. 1. The aim of the current study is to provide a solution to segmentation of fine processes associated with filopodia and lamellipodia, which would help pinpoint the physical interaction points between neural stem cells during this critical period of development when cell type is being defined.

Detection and segmentation of cells in microscopic images have been extensively studied. However, most existing methods exclusively focus on either cell detection or segmentation. Although a few of them attempt to address both the two tasks, they generally treat the detection and segmentation separately with multiple stages. To name a few, Althoff et al. (2005) detect the neural stem cells with a multi-scale Laplacian of Gaussian (LoG) filter, and then segment each cell via dynamic programming. Peng et al. (2009) localize the stem cells using multi-scale blob and curvilinear structure detectors, and then delineate each cell with multi-level sets. Wu et al. (2015) detect cells via greedy search and then obtain the boundary of each cell using active contour. The above-mentioned unsupervised methods are sensitive to intensity variations, and many efforts would be required to adjust their parameters for each dataset.

Instance segmentation is a task combining both object detection and segmentation. Recently, this topic has received much attention along with the development of supervised deep neural network (DNN) techniques. DNN methods have achieved remarkable

* Corresponding author.
E-mail addresses: jy486@cs.rutgers.edu (J. Yi), pw241@cs.rutgers.edu (P. Wu), menglin.jiang@cs.rutgers.edu (M. Jiang), qh55@cs.rutgers.edu (Q. Huang), daniel.hoeppner@astellas.com (D.J. Hoeppner), dnm@cs.rutgers.edu (D.N. Metaxas).

https://doi.org/10.1016/j.media.2019.05.004
1361-8415/

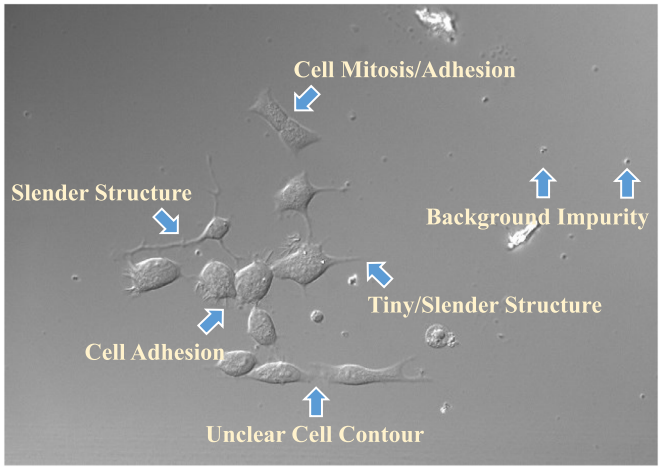


Fig. 1. Illustration of challenges in neural cell instance segmentation. Neural cells often have variant shapes and appearances, tiny and slender structures, occlusion, and obscure contours. Microscopic images usually have low contrast and background impurities.

Table 1
Specific parameters of the proposed network. Residual mapping (He et al., 2016) is performed in conv2 to conv4. n is 6 and 23 for ResNet50 and ResNet101, respectively. ReLU (Nair and Hinton, 2010) and batch normalization (Ioffe and Szegedy, 2015) are used in hidden layers.

Layer	Output size	ResNet-based SSD
conv1	256 × 256	7 × 7, 64, stride 2
conv2_x	128 × 128	$\begin{Bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{Bmatrix} \times 3$
conv3_x	64 × 64	$\begin{Bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{Bmatrix} \times 4$
conv4_x	32 × 32	$\begin{Bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{Bmatrix} \times n$
conv5_x	16 × 16	$1\times1, 256$ $3\times3, 512$
conv6_x	8 × 8	$1\times1, 128$ $3\times3, 256$

performance in many computer vision problems, such as object detection (Girshick et al., 2014; Girshick, 2015; Ren et al., 2015; Redmon et al., 2016; Liu et al., 2016; Fu et al., 2017) and semantic segmentation (Long et al., 2015; Noh et al., 2015; Yang et al., 2018). Based on these single-task networks, several DNN models were proposed for instance segmentation, such as MNC (Dai et al., 2016), FCIS (Li et al., 2017), Mask R-CNN (He et al., 2017), and MaskLab (Chen et al., 2018). Existing instance segmentation methods are mainly based on two-stage object detector, at the core of which is a region proposal network (RPN) (Ren et al., 2015). To perform segmentation, they generally adopt the deep feature maps that contain high-level semantics, while ignoring the shallow ones which are rich in low-level shape and texture information of the objects. Consequently, these methods are insufficient for capturing details, such as the tiny and slender structures of neural cells.

To overcome the drawbacks mentioned above, we propose an attentive instance segmentation model that is able to accurately capture the neural cell instances at a fast speed. The overview of our approach is shown in Fig. 2. In particular, our method employs a single shot multi-box detector (SSD) (Liu et al., 2016) to detect neural cells in the input image. To improve detection accuracy and speed, we propose two strategies. First, we employ a feature fusion module, which consolidates shallow (fine) and deep (coarse) feature maps to facilitate the detection of small cells. Second, we incorporate an attention module in SSD to help it focus on useful image regions while suppressing the irrelevant background information. With the bounding box predictions from SSD, we then crop the cell instance regions accordingly from multi-scale feature maps and pass them into the mask prediction module. To perform the cell segmentation, we build a U-net (Ronneberger et al., 2015) that shares the backbone layers with SSD. The U-net propagates semantics from the deep layers to the shallow ones through a skip connection. To highlight useful regions and suppress the noisy information, we design and incorporate an attention module to the skip connection.

The proposed model is fast and accurate. It is capable of capturing the tiny and slender structures of neural cells. This paper makes several contributions. (1) Different from the state-of-the-art instance segmentation networks, the proposed model builds upon a one-stage object detector SSD and inherits its fast speed. (2) Existing CNN-based instance segmentation networks rely on ROI pooling or ROI aligning strategy, which samples a cropped region to a small fixed size from a particular deep feature map. This strategy loses details of neural cells. In contrast, our segmentation network combines multi-scale feature maps and therefore is able to capture the tiny and slender structures of neural cells. (3) We design two different attention units to respectively improve the accuracy of neural cell detection and segmentation.

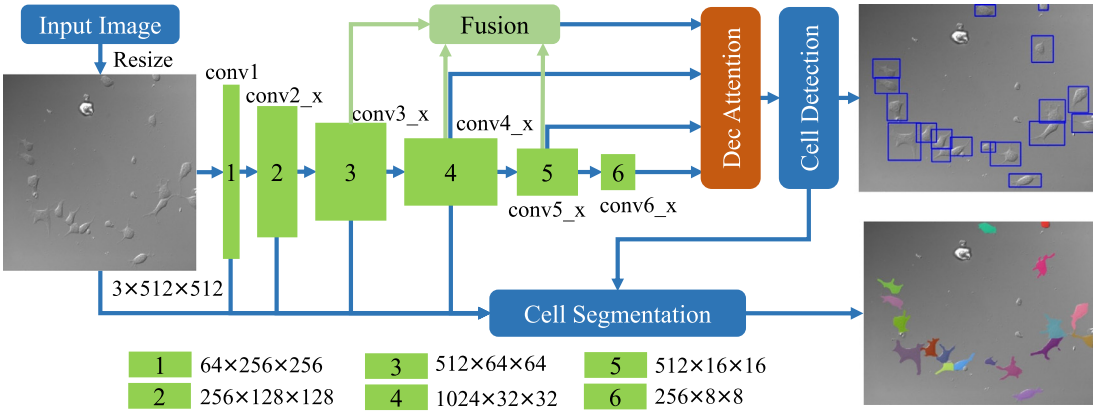


Fig. 2. Overview of the proposed model. The input image is resized to 512 × 512 before being fed into the network. Sizes of the input image and feature maps are displayed in the format of "number of channels × height × width". Note that the input images are in grayscale. The reason that we use three-channel inputs is to take advantage of the pre-trained weights of the backbone networks. The symbol "_x" represents multiple convolutional layers. The specific settings of the convolutional layers can be found in Table 1.

This paper is an extension of our preliminary works (Yi et al., 2018a, 2018b) with several modifications. First, we introduced two kinds of attention mechanisms to enhance the accuracy of both detection and segmentation. Second, we added ablation studies, more comparison experiments and mathematical formulations.

The rest of this paper is organized as follows. Section 2 reviews relevant works of detection, segmentation, and instance segmentation. Section 3 describes the proposed attentive neural cell instance segmentation approach. Section 4 presents the experimental results, including the ablation study, comparison experiments. Finally, Section 5 concludes this paper.

2. Related work

In this section, we first summarize some current cell detection and segmentation methods. We also review the state-of-the-art deep learning based detection and segmentation methods. Then, we review the development of instance segmentation techniques. Finally, we briefly introduce the attention models.

2.1. Cell detection

There are numerous works in literature studying cell detection in microscopic images. Blob detection (Althoff et al., 2005; Peng et al., 2009; Al-Kofahi et al., 2010), seeded watershed (Vincent and Soille, 1991; Pinidiyaarachchi and Wählby, 2005), graph partition (Boykov and Jolly, 2001; Yang et al., 2008; Bernardis and Yu, 2010; Zhang et al., 2014) are frequently used to locate the cells. Blob detection uses filters such as Laplacian of Gaussian (LoG) to find the locations of the cells, and the parameters and scales of the filters need to be carefully tuned in order to achieve good performance. Seeded watershed detects cell boundaries according to the local maxima intensities (gradients). Graph partition methods are sensitive to intensity variation between cells and are computationally expensive. The above-mentioned unsupervised methods rely on heuristics, and thus fail to generalize well to different microscopic images. Early supervised cell detection methods often employ hand-crafted features, including SIFT (Lowe, 2004), HOG (Dalal and Triggs, 2005), Haar feature (Lienhart and Maydt, 2002), in conjunction with traditional classifiers, such as support vector machine (SVM) (Cortes and Vapnik, 1995; Wang et al., 2008), random forest (Breiman, 2001; Kainz et al., 2015), and AdaBoost (Freund and Schapire, 1997; Vink et al., 2013). However, these hand-crafted features, which characterize color, intensity, texture, or shape, are not sufficient to describe the appearance of neural cells.

In recent years, methods based on convolutional neural networks (CNNs) have exhibited remarkable performance in object detection. The pioneering works, including R-CNN (Girshick et al., 2014), fast R-CNN (Girshick, 2015), and faster R-CNN (Ren et al., 2015) have achieved significant success and become the most popular two-stage object detectors. Specifically, the first stage of these methods is to extract the region proposals, and the second stage is to use ROI pooling or ROI alignment to up-sample the proposal regions into fixed size (e.g., 7×7). These methods have been applied to cell-related tasks, such as mitosis detection (Li et al., 2018) and lymphocyte detection (Garcia et al., 2017). While being effective, the two-stage object detection methods are quite computationally expensive. To address this issue, the one-stage object detectors, including YOLO (Redmon et al., 2016), SSD (Liu et al., 2016; Yi et al., 2017), DSSD (Fu et al., 2017), are developed. The one-stage detectors discard the region proposal stage, and integrate the box regression and class prediction into the same feedforward network. Among these methods, SSD achieves a good tradeoff between accuracy and speed. However, since SSD is built upon the heavy VGG network, it is still not sufficiently efficient for processing large video datasets.

2.2. Cell segmentation

Traditional cell segmentation methods generally rely on models whose parameters need to be carefully tuned. Examples include simple thresholding (Otsu, 1979; Sahoo et al., 1988; Sankaran and Asari, 2006), watershed (Vincent and Soille, 1991; Vincent, 1993; Koyuncu et al., 2012), graph cut (Boykov and Jolly, 2001; Bensch and Ronneberger, 2015), and active contour (Kass et al., 1988; Wu et al., 2015). In particular, simple thresholding may fail when foreground and background have diffuse transition. Besides, it is unable to separate attached cells, and typically requires post-processing to remove false positive noise. Region growing-based methods, such as watershed, are sensitive to intensity variation and therefore suffer from over-segmentation. Graph cut models segment cells that are different in appearance from their neighbors; however, such property makes them less robust to the intensity variation. Active contour is sensitive to the initial segmentation and model parameters, and usually suffers from early boundary stop and boundary leakage.

In recent years, CNN-based object segmentation methods have gained increased popularity. Long et al. (2015) introduce a pioneering fully convolutional network (FCN) that achieves an end-to-end and pixel-wise semantic segmentation. In particular, they propose a skip connection that combines the deep coarse feature maps with the shallow fine ones through bilinear interpolation, which largely improves the precision and robustness of semantic segmentation. Instead of using bilinear interpolation, Noh et al. (2015) propose to learn a deconvolutional layer for semantic segmentation. Ronneberger et al. (2015) take a step further and combine the deconvolutional layer with the skip connection to build a “U” shaped network (termed U-net), which improves the segmentation performance to a large extent. These CNN-based segmentation methods demonstrate outstanding performance compared with traditional unsupervised methods, and have been extensively applied to medical and biological image processing, such as histological cell segmentation (Chen et al., 2017) and abdominal aortic thrombus segmentation (López-Linares et al., 2018). In this paper, we adopt the U-net to perform accurate cell segmentation.

2.3. Instance segmentation

Inspired by the success of DNN techniques in both object detection and segmentation, researchers have recently proposed the task of instance segmentation, which aims at joint object detection and segmentation. The state-of-the-art instance segmentation methods, such as MNC (Dai et al., 2016), FCIS (Li et al., 2017) and Mask R-CNN (He et al., 2017) have achieved impressive results in natural image instance segmentation. However, these methods are based on two-stage detectors, and thus are computationally inefficient. Besides, they predict the object segmentation map based on the last feature map and ignore the rich low-level details of the objects contained within shallow layers. Another different approach, DCAN (Chen et al., 2017), discards the object detector completely, and predicts the cell segmentation masks and contours jointly via a single unified CNN. The two prediction results are then combined to produce a contour-aware cell instance segmentation. However, DCAN also fails to consider the low-level details from the shallow layers, and its performance heavily depends on the quality of the predicted cell boundaries. As a result, DCAN would fail to separate attached cells when the boundary is fuzzy.

Due to the inherent design, the above-mentioned methods are not suitable for predicting the fine details of objects. To improve the detection and segmentation of the tiny and slender structures

of neural cells, we propose to combine the shallow and deep features. We verify that our method is efficient and accurate compared to existing neural cell detection and segmentation methods.

2.4. Attention mechanism

Attention mechanism is motivated by human visual systems and has been applied to DNN for helping the model focus on useful information while suppressing the irrelevant information. In particular, attention models have been extensively studied and utilized in language translation (Vaswani et al., 2017), object detection (Hu et al., 2018), video classification (Wang et al., 2018), and semantic segmentation (Chen et al., 2016; Zhang et al., 2018a). It has been demonstrated that the attention models are quite powerful in relating long-range dependencies across channels and spatial positions. In this work, we propose two kinds of attention units. The first one is used to help the network focus on important regions during cell detection, and the other one is employed to suppress false segmentation results. Both units carry a global context of the feature maps, and thus is able to guide the detection and segmentation effectively.

3. Methods

An overview of the proposed network is given in Fig. 2. The proposed model is a unified and end-to-end trainable network that simultaneously performs cell detection and segmentation. The input image is resized to 512×512 before being fed into the network. The convolutional layers, from conv2 to conv4, are residual networks (He et al., 2016). The specific settings of the convolutional layers are listed in Table 1. Below we introduce our neural cell detection and segmentation modules in details.

3.1. Cell detection

We demonstrate the cell detection module in Fig. 2. The feature maps from conv3, conv4, and conv5 are sent into a fusion module. The fusion module combines the shallow and deep semantics and outputs a feature map that replaces the feature map from conv3, with a desire to improve the detection accuracy of small cells. Then the detection attention module is applied to feature maps from conv3-6 to recalibrate the features and highlight the useful regions on the feature map. These feature maps are then gathered for bounding box and confidence prediction. In particular, the shallow and deep layers are responsible for the detection of small and large cells, respectively.

3.1.1. Feature fusion module

To integrate the complementary descriptive power of shallow and deep layers, we fuse the feature maps of conv3, conv4, and conv5, and use the fused feature map to replace the output of conv3. In this way, our method could recognize small neural cells more precisely (see Table 2). The structure of our feature fusion module is shown in Fig. 3. In particular, we first transfer the three feature maps to the same channel size (we use 256 in this paper) through a single 1×1 convolutional layer. The feature maps from conv4 and conv5 are then resized using bilinear interpolation so that they have the same size of 64×64 as feature map from conv3. Next, the feature maps from conv3-5 are concatenated. Finally, the concatenated feature map is transformed to the same channel size of conv3 through a 1×1 convolutional layer.

3.1.2. Detection attention module

We introduce the attention mechanism (Vaswani et al., 2017; Zhang et al., 2018b; Wang et al., 2018) to the detection module to refine the feature maps and make the model pay attention to useful regions. The attention mechanism calculates the responses of a position as a weighted sum of the features at all the positions. We employ the detection attention module to highlight the cell regions on feature maps from conv3-6 with a desire to make further improvement of cell detection accuracy.

The detection attention module is shown in Fig. 4. Suppose $\mathbf{x}_l \in \mathbb{R}^{C_l \times N}$ is an input feature map, where $l \in \{3, 4, 5, 6\}$ indicates the layer of feature map, C is the channel numbers, $N = W \times H$ is the total number of spatial pixels at each channel. We first transfer the input feature map $\mathbf{x} \in \mathbb{R}^{C \times N}$ to feature spaces $\mathbf{q}$, $\mathbf{k}$, $\mathbf{v}$ through

$$\mathbf{q}(\mathbf{x}) = \mathbf{W}_q \mathbf{x}, \ \mathbf{W}_q \in \mathbb{R}^{C' \times C}, \tag{1}$$

$$\mathbf{k}(\mathbf{x}) = \mathbf{W}_k \mathbf{x}, \ \mathbf{W}_k \in \mathbb{R}^{C' \times C}, \tag{2}$$

$$\mathbf{v}(\mathbf{x}) = \mathbf{W}_v \mathbf{x}, \ \mathbf{W}_v \in \mathbb{R}^{C \times C}, \tag{3}$$

where $C' = C/8$. The feature $\mathbf{q}(\mathbf{x}), \mathbf{k}(\mathbf{x}), \mathbf{v}(\mathbf{x})$ play the role of query, key and value in language transformer (Vaswani et al., 2017). Note that we squeeze the channels of $\mathbf{q}(\mathbf{x})$ and $\mathbf{k}(\mathbf{x})$ in order to save the computation time. The $\mathbf{q}(\mathbf{x})$ and $\mathbf{k}(\mathbf{x})$ are from $\mathbf{x}$ itself, so we

Table 2
Effects of backbone networks, feature fusion module, and detection attention module. AP_{box} is evaluated using PASCAL VOC 2007 metric. Time (seconds) is calculated on a single NVIDIA K40 GPU.

BaseNet	AP_{box}@0.5	AP_{box}@0.7	Time
VGG16	67.99	23.02	0.1481
VGG16-Fusion	69.06	30.92	0.1537
VGG16-Fusion-Attn	76.10	30.98	0.1721
ResNet50	77.59	32.59	**0.0563**
ResNet50-Fusion	78.22	33.98	0.0612
ResNet50-Fusion-Attn	**79.55**	**35.92**	0.0807
ResNet101	67.57	25.16	0.0968
ResNet101-Fusion	77.54	34.65	0.1018
ResNet101-Fusion-Attn	77.86	33.91	0.1220

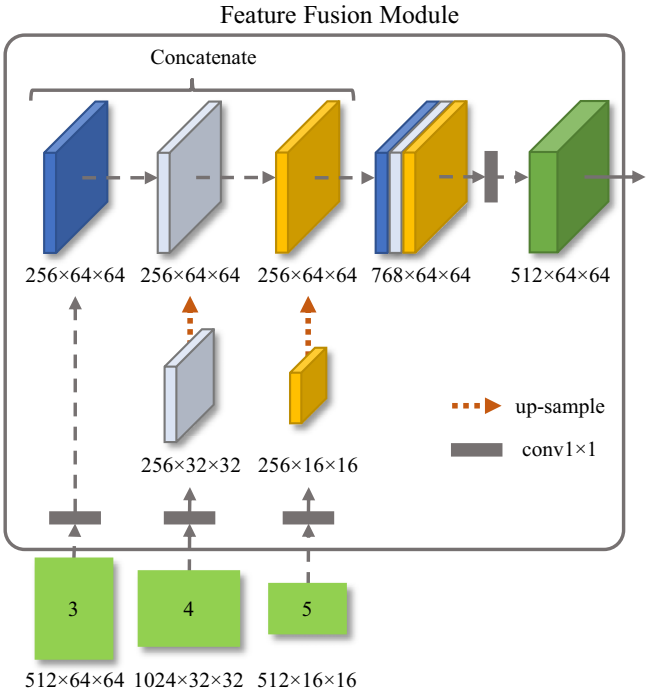


Fig. 3. The feature fusion module. Sizes of the feature maps are displayed in the format of "number of channels × height × width".

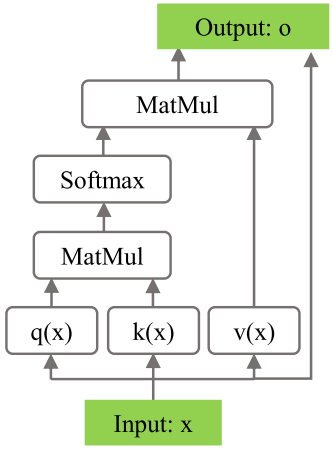


Fig. 4. Detection attention module. MatMul represents matrix multiplication.

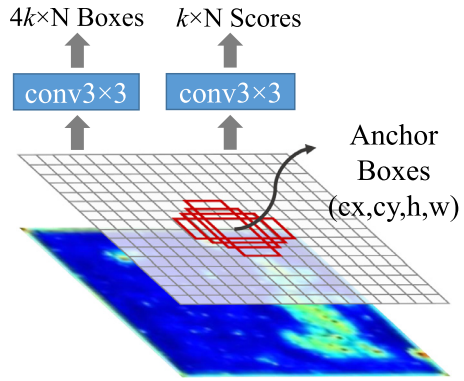


Fig. 5. Detection module. Input feature maps are transferred to two branches by 3×3 convolutional layers. One branch is for box regression. The other branch is for box confidence prediction. k denotes the number of anchor boxes at each grid, N is the total number of grids.

also call the attention mechanism as self-attention. Now we can create an attention map by applying matrix multiplication of $\mathbf{q}(\mathbf{x})$ and $\mathbf{k}(\mathbf{x})$, and we normalize it with a softmax operation:

$$a_{ij} = \frac{\exp(z_{ij})}{\sum_{j=1}^{N} \exp(z_{ij})}, \ i, j = 1, 2, \cdots, N, \tag{4}$$

$$\mathbf{z} = \mathbf{q}(\mathbf{x})^{\top}\mathbf{k}(\mathbf{x}), \ \mathbf{z} \in \mathbb{R}^{N \times N}. \tag{5}$$

The element a_{ij} of the attention map $\mathbf{a} \in \mathbb{R}^{N \times N}$ describes the extent the model considers position j when position i of a feature map is queried. Next, for each channel c, the attention map $\mathbf{a}$ is used as a weight map that weighted-sums the values at every spatial position of $\mathbf{v}(\mathbf{x})$ to represent the cell properties at that position. $\mathbf{v} \in \mathbb{R}^{C \times N}$ is a variant of the input feature map $\mathbf{x} \in \mathbb{R}^{C \times N}$. Then the output feature map $\mathbf{o} \in \mathbb{R}^{C \times N}$ of the detection attention map is

$$\mathbf{o} = \mathbf{x} + \mathbf{v}(\mathbf{x})\mathbf{a}^{\top}. \tag{6}$$

3.1.3. Bounding box regression and confidence prediction

The output feature maps from the attention module conv3-6 are input into the cell detection module for bounding box regression and confidence prediction. The detection strategy is the same as SSD (Liu et al., 2016). First, SSD spreads the anchor boxes to multi-scale feature maps densely. Then, SSD encodes the ground-truth boxes and labels to the anchors and create the encoded anchors and their corresponding labels. Finally, the predictions and the encoded anchors and labels are used by loss function to optimize the model weights.

Anchor box assignment: To assign the anchor boxes, SSD discretizes the input feature maps, namely conv3-6, into 1×1 grids (see Fig. 5). Each grid is regarded as the center of the anchor boxes, with different aspect ratio and scale for different feature maps. In particular, the lower feature map is responsible for detecting cells with a smaller scale as the lower feature maps have more fine details than higher feature maps. Therefore, the scales of the anchor boxes are increasing from conv3 to conv6. Specifically, we use aspect ratios (0.5,2), (0.5, 2, 1/3, 3), (0.5, 2, 1/3, 3), (0.5, 2, 1/3, 3) and scales 0.04, 0.1, 0.26, 0.42 for conv3, conv4, conv5, and conv6, respectively.

Encoding ground-truth: Next, the ground-truth bounding box of each cell is encoded to the anchors. The encoding process is as follows: suppose the locations of the anchors are (cx_a, cy_a, w_a, h_a) and the ground-truth locations are (cx_g, cy_g, w_g, h_g), where cx and cy are the center of the box, w and h refer to the width and height of the box. The intersection of union (IoU) is used to pair each anchor box and ground-truth box. We assign the positive labels to anchors that have IoU greater than 0.5, and assign the negative labels to the remaining anchors. The locations of the encoded anchors ($g = (cx_{a_{\text{new}}}, cy_{a_{\text{new}}}, w_{a_{\text{new}}}, h_{a_{\text{new}}})$) are updated with the offsets between the anchors and their best-matched ground-truth boxes (Girshick et al., 2014; Ren et al., 2015):

$$\begin{aligned} cx_{a_{\text{new}}} &= (cx_a - cx_g)/w_a \\ cy_{a_{\text{new}}} &= (cy_a - cy_g)/h_a \\ w_{a_{\text{new}}} &= \log(w_a/w_g) \\ h_{a_{\text{new}}} &= \log(h_a/h_g). \end{aligned} \tag{7}$$

Predicted features: As shown in Fig. 5, the input feature maps (conv3-6) are transferred to the box regression and confidence prediction branches through two 3×3 convolutional layers. Suppose an input feature map is $\mathbf{x} \in \mathbb{R}^{C \times N}$, where C is the feature channel number, $N = W \times H$ is the total number of spatial pixels at each channel. The two branch output features are $\mathbf{x}_{box} \in \mathbb{R}^{4k \times N}$ and $\mathbf{x}_{conf} \in \mathbb{R}^{k \times N}$, where k is the number of anchors we assign to each grid.

3.1.4. Detection loss

The loss function of the detection module includes two parts: bounding box and confidence. Therefore, the total detection loss function is a weighted combination form:

$$L_{\text{det}} = \frac{1}{N_{\text{pos}}}(L_{\text{locs}} + \alpha L_{\text{conf}}), \tag{8}$$

where α is a weight factor, N_{pos} is the number of positive anchors, L_{locs} and L_{conf} are bounding box loss and confidence loss, respectively. The form of L_{locs} is a smooth L_1 loss (Girshick et al., 2014; Liu et al., 2016):

$$L_{\text{locs}} = \sum_{i \in \text{pos}} \sum_{m \in \{cx,cy,w,h\}} \text{smooth}_{L_1}(l_i^m - g_i^m), \tag{9}$$

$$\text{smooth}_{L_1}(z) = \begin{cases} 0.5z^2 & \text{if } |z| < 1 \\ |z| - 0.5 & \text{otherwise} \end{cases}, \tag{10}$$

where l_i^m and g_i^m refer to the predicted boxes and encoded anchor boxes, respectively, m indicates the coordinate parameters. L_{conf} is a binary cross entropy loss between the ground-truth confidence and the predicted box confidence:

$$L_{\text{conf}} = -\sum_i (x_i \log p_i + (1 - x_i)\log(1 - p_i)) \tag{11}$$

where x_i is the labeled confidence, and p_i is the predicted confidence, i refers to each position in a segmentation mask.

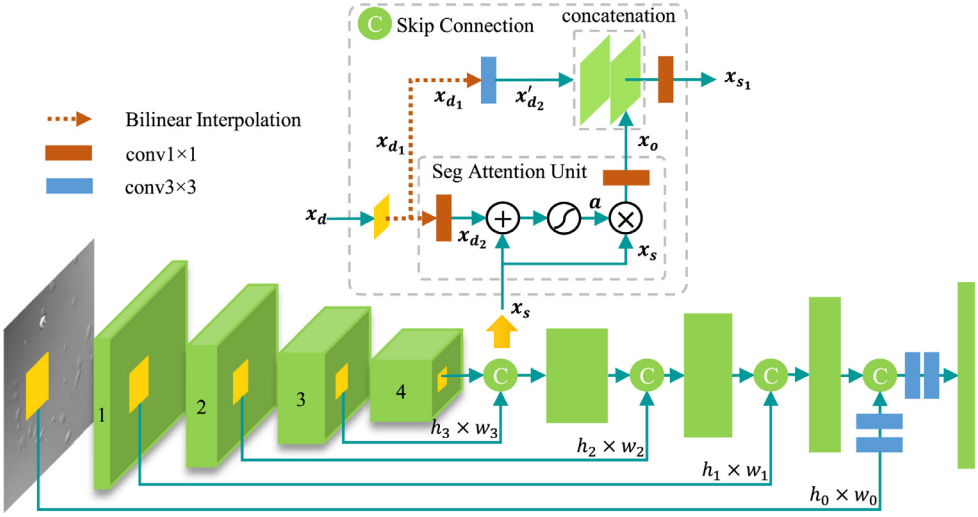


Fig. 6. Segmentation module. Skip connection integrates a shallow (fine) layer with a deep (coarse) layer. Segmentation attention unit is used to suppress noise features.

3.2. Cell segmentation

The cell segmentation module is illustrated in Fig. 6. According to the bounding box prediction results, we crop the instance cell regions from feature maps from conv1 to conv4 and utilize these cropped regions for instance cell segmentation. We term these regions as ROI regions. Different from the existing state-of-the-art instance segmentation methods (Dai et al., 2016; Li et al., 2017; He et al., 2017) that only employ the instance ROI regions from a fix-scale feature map, the proposed model combines multi-scale feature maps with a desire to make a precise segmentation prediction.

The segmentation module utilizes a similar strategy as FCN (Long et al., 2015) and U-net (Ronneberger et al., 2015), where a skip connection combines a shallow layer with a deep layer. However, we find such a connection is weak in suppressing the background noises (see Fig. 9). Therefore, we introduce a segmentation attention unit in the skip connection to reduce the false predictions.

3.3. Skip connection

As illustrated in Fig. 6, the skip connection takes in a deep feature map $\mathbf{x}_d \in \mathbb{R}^{C_d \times W_d \times H_d}$ and a shallow feature map $\mathbf{x}_s \in \mathbb{R}^{C_s \times W_s \times H_s}$, where *C, W, H* is the channel size, height and width of a feature map. There are two branches in the skip connection module. One is concatenation branch, another one is the segmentation attention unit branch. First, the deep feature map $\mathbf{x}_d$ is upsampled to $\mathbf{x}_{d_1} \in \mathbb{R}^{C_d \times W_s \times H_s}$ through a bilinear interpolation, where $\mathbf{x}_{d_1}$ has the same size as the shallow feature map $\mathbf{x}_s$. Then $\mathbf{x}_{d_1}$ is passed into the segmentation attention unit and the concatenation branch. The output feature of the skip connection $\mathbf{x}_{s_1}$ is generated from the concatenation branch.

3.3.1. Segmentation attention unit

The segmentation attention unit first transfers $\mathbf{x}_{d_1}$ to $\mathbf{x}_{d_2} \in \mathbb{R}^{C_s \times W_s \times H_s}$ through a 1×1 convolutional layer, where $\mathbf{x}_{d_2}$ has the same channel size as $\mathbf{x}_s$. Then we calculate the summation of $\mathbf{x}_{d_2}$ and $\mathbf{x}_s$ by $\mathbf{s} = \mathbf{x}_{d_2} + \mathbf{x}_s$. The feature $\mathbf{s}$ contains semantics from both the deep feature map and the shallow feature map. As a deep feature map contains high-level semantics, it would be helpful for suppressing the background noises in the shallow feature map. Next, we normalize $\mathbf{s}$ through a Sigmoid operation to generate the attention map $\mathbf{a} \in \mathbb{R}^{C_s \times W_s \times H_s}$:

$$a_{c,i,j} = \frac{1}{1 + \exp(-s_{c,i,j})}, \ c = 1, 2, \cdots C_s, \ i = 1, 2, \cdots W_s, \ j = 1, 2, \cdots H_s, \tag{12}$$

where $a_{c,i,j}$ indicates the possibility that this pixel should be considered. Finally, we use the feature map $\mathbf{a}$ as a weight map that highlights the useful pixels in the shallow layer. The output of the segmentation attention unit is $\mathbf{x}_o$:

$$\mathbf{x}_o = \mathbf{W}_s(\mathbf{a} \cdot \mathbf{x}_s) + \mathbf{b}_s \tag{13}$$

where $\mathbf{W}_s \in \mathbb{R}^{C_s \times C_s}$, $\mathbf{b}_s \in \mathbb{R}^{C_s}$. The linear transformation is implemented through a 1×1 convolutional layer.

3.3.2. Concatenation

The concatenation branch takes in $\mathbf{x}_{d_1}$ and $\mathbf{x}_o$. First, $\mathbf{x}_{d_1}$ is transferred to $\mathbf{x}'_{d_2} \in \mathbb{R}^{C_s \times W_s \times H_s}$ through a 3×3 convolutional layer. Now $\mathbf{x}'_{d_2}$ has the same channel size as $\mathbf{x}_o$. Next, we concatenate $\mathbf{x}'_{d_2}$ and $\mathbf{x}_o$, and we transfer the concatenated feature to the output feature $\mathbf{x}_{s_1} \in \mathbb{R}^{C_s \times W_s \times H_s}$ with a 1×1 convolutional layer.

3.4. Segmentation loss

The objective loss of the segmentation is a binary-cross entropy loss:

$$L_{\text{seg}} = -\frac{1}{N}\sum_{j=1}^{N}\frac{1}{M_j}\sum_{i=1}^{M_j}(t_{ij}\log p_{ij} + (1 - t_{ij})\log(1 - p_{ij})), \tag{14}$$

where p and t are respectively the predicted and ground-truth segmentation probability map. i indexes the positions of pixels in the map, j indexes positive predicted bounding box, M is the total number of pixels in the segmentation map, and N is the total number of positive predicted bounding boxes.

4. Experimental results

4.1. Data description

We sample 644 neural cell images from a collection of time-lapse microscopic videos of rat CNS stem cells (Ravin et al., 2008), where the image size is 640×512. The neural cell data is captured using the Nomarski DIC optics with a 40x oil NA 1.3 lens

and the ORCA ER CCD camera which captures 16-bit grayscale with 2×2 binning at final 640×512 px at 1.4 um/px. Images are updated every 2 min. We randomly select 386 images for training, 129 for validation, and 129 for testing. Our annotations are human-generated without any machine assistance. There are three experts involved in the annotation process. The accuracy of the annotation is validated by two other experts. We use Adobe Photoshop CC to create the annotations. In particular, each cell in the input image is labeled with a particular color, as shown in Fig. 9. Sequential frames are utilized to validate the ambiguous pixels. To reduce over-fitting, we adopt data augmentation, transfer learning, and validation strategies. The data augmentation includes random expanding, cropping, flipping, contrast distortion and brightness distortion to the training images to make the model more robust to various input cell sizes and shapes. We transfer the weights of backbone networks that are pre-trained on ImageNet dataset to our model to improve the task performance on a smaller dataset.

4.2. Evaluation metrics

We evaluate the instance segmentation accuracy by mask-level average precision (AP) (Everingham et al., 2010) at IoU (intersection-over-union) threshold of α, following the existing works (Dai et al., 2016; Li et al., 2017; He et al., 2017; Yi et al., 2018a). We term the metric as AP@α. AP is defined as the mean precision (p) at 11 recall (r) levels ({0, 0.1, $\cdots$, 1}) (Everingham et al., 2010):

$$AP = \frac{1}{11} \sum_{l \in \{0, 0.1, \cdots, 1\}} \max(p(r \geq l)). \tag{15}$$

To calculate the AP@α, we collect the predictions of the proposed model, each prediction involves a bounding box, a confidence score, and a segmentation mask. Non-maximum-suppression (NMS) is applied to suppress repetitive predictions. Next, we sort predictions according to their confidence scores in a descending order to ensure the predictions with high scores are considered first. For each prediction, we calculate its maximum IoU between the predicted segmentation mask and all the ground-truth masks:

$$\text{IoU} = \max_i \left(\frac{\mathbf{x} \cap \mathbf{y}_i}{\mathbf{x} \cup \mathbf{y}_i} \right). \tag{16}$$

where $\mathbf{x} \in \mathbb{R}^{W \times H}$ is the predicted segmentation mask, $\mathbf{y}_i \in \mathbb{R}^{W \times H}$ is the ith ground-truth mask, W and H are the image width and height. A prediction is considered as a true positive instance if the maximum IoU is greater than a threshold α, the according ground-truth is marked as detected. Any repetitive detection will be considered as a false-positive instance. Finally, the AP metric summarizes the shape of the precision/recall curve and gives an evaluation measured both instance detection and segmentation accuracy.

In addition to AP@α, we also measure the average IoU at thresholds α. To be clear, we define

$$\text{AIoU@}\alpha = \frac{1}{N_\alpha} \left(\sum_{i=1}^{N_\alpha} (\text{IoU})_i \right) \tag{17}$$

where N_α is the total number of predictions that satisfy $\text{IoU} \geq \alpha$.

The detection evaluation metric is the box-level average precision AP_{box}@α, where the IoU is calculated between the predicted bounding box and the ground-truth bounding box (Everingham et al., 2010).

4.3. Implementation details

Our method is implemented with Pytorch and NVIDIA K40 GPUs. The backbone networks are fine-tuned with the weights pre-trained on ImageNet (Deng et al., 2009), other parts of the network are initialized with weights sampled from a standard Gaussian distribution. To accelerate the training process, we train the detection and segmentation modules separately. In particular, the weights of the detection module are frozen when training the segmentation module. Moreover, we use stochastic gradient descent (SGD) with a mini-batch size 64 to optimize the hyper-parameters. The initial learning rate of the SGD is 0.001. We use a weight decay of 0.0005 and a momentum of 0.9. The training and validation losses of the detection module and the segmentation module are exhibited in Figs. 7 and 8, respectively. From Figs. 7 and 8 we can observe that our model avoids over-fitting, thanks to the data augmentation, validation and transfer learning.

4.4. Ablation study for cell detection

The ablation study for different backbone networks with and without feature fusion module and detection attention module is exhibited in Table 2. From Table 2 we can observe that the fusion module and attention module make the detection more accurate. Moreover, the model with ResNet50 backbone achieves the highest accuracy and fastest speed compared to the other two backbones, especially when attention module is used. The reason that the ResNet50 achieves the fastest speed is probably because ResNet-50 contains 25.5×10^6 parameters (Xie et al., 2017), while ResNet-101 contains 44.7×10^6 parameters and VGG16 contains 133×10^6 parameters (Simonyan and Zisserman, 2015). The results demonstrate that ResNet50 is more suitable for the neural cell dataset.

We also investigate the effects of conv6 and input channels for ResNet50 backbone with the feature fusion module and detection attention module. The results are shown in Table 3, which suggest that conv6 is necessary to detect cells of relatively large size. It also reveals that the pre-trained weights for 3-channel input images are essential to improve the detection performance on our small dataset.

4.5. Instance segmentation results

From the ablation study above, we observe that ResNet is better than VGG16 in both accuracy and speed. Therefore, we report our instance segmentation results with ResNet backbone. In particular, we compared our method with three state-of-the-art instance segmentation algorithms, namely DCAN (Chen et al., 2017), MNC (Dai et al., 2016), FCIS (Li et al., 2017) and Mask R-CNN (He et al., 2017) and one of the most recent automatic neural stem cell detection and segmentation method IMDLMS (Peng et al., 2009). The IMDLMS is evaluated under the same parameter settings.

4.5.1. Quantitative results

The quantitative cell instance segmentation results are shown in Table 4. It can be observed that the unsupervised method IMDLMS (Peng et al., 2009) is time-consuming and inaccurate, while the other supervised methods are more accurate and fast. Although DCAN (Chen et al., 2017) runs the fastest among all the evaluated systems, it suffers from unsatisfactory detection and segmentation accuracy. Except for DCAN, the proposed model outperforms the other state-of-the-art methods in both accuracy and

Table 3
Effects of conv6 and input channels on detection performance.

BaseNet	Include conv6?	Input channels	AP_{box}@0.5	AP_{box}@0.7
ResNet50-Fusion-Attn	×	3	78.00	33.60
ResNet50-Fusion-Attn	✓	1	77.52	33.45
ResNet50-Fusion-Attn	✓	3	**79.55**	**35.92**

Table 4
Results of neural cell instance segmentation. Time (seconds) is calculated on a single NVIDIA K40 GPU.

Method	AP@0.5	AP@0.7	AIoU@0.5	AIoU@0.7	Time
IMDMLS (Peng et al., 2009)	10.83	2.91	56.83	71.05	48.00
DCAN Chen et al. (2017)	25.91	2.27	59.21	72.31	**0.0041**
MNC (Dai et al., 2016)	48.72	11.37	62.73	75.47	0.4750
FCIS (Li et al., 2017)	66.02	7.13	64.85	75.07	0.2130
Mask R-CNN (He et al., 2017)	59.94	25.87	72.10	79.30	0.7486
Ours-ResNet50	88.89	63.34	77.55	80.12	0.1575
Ours-ResNet50-Attn	**89.06**	**70.96**	77.75	80.30	0.1753
Ours-ResNet101	88.74	63.10	77.57	80.60	0.2021
Ours-ResNet101-Attn	88.88	64.92	**77.98**	**80.83**	0.2201

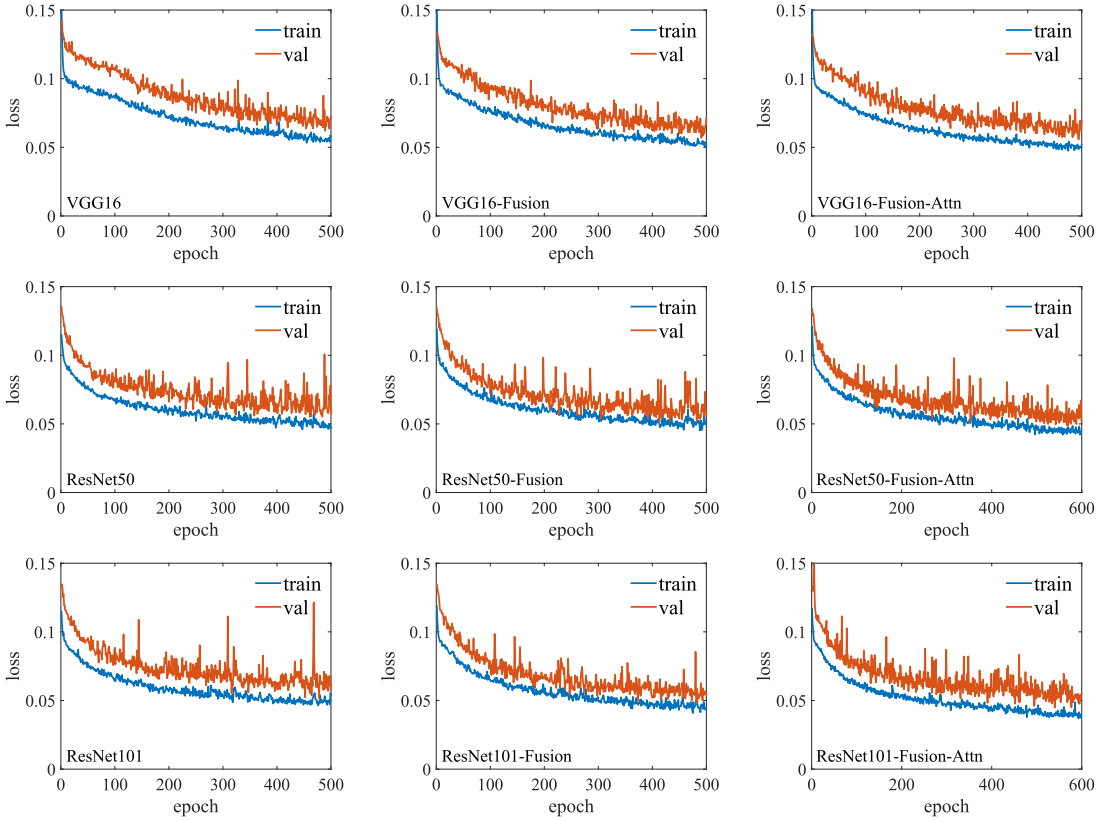


Fig. 7. Training and validation loss values of neural cell detection.

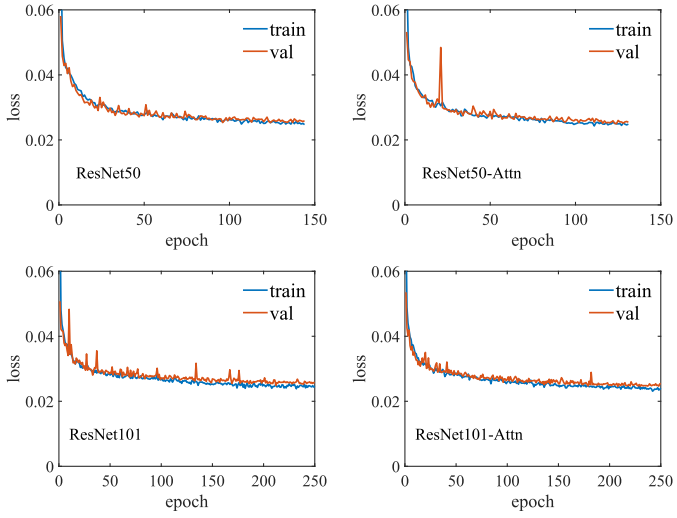


Fig. 8. Training and validation loss values of neural cell instance segmentation.

speed. Especially for ResNet50 backbone, our model achieves the highest accuracy and the fastest speed. Even for ResNet101 backbone, the proposed model only takes about 0.2s on NVIDIA K40 GPU. The fastest speed in our model would be owing to the strategy that we employ the one-stage detector with a ResNet backbone, while the other three state-of-the-arts, MNC (Dai et al., 2016), FCIS (Li et al., 2017) and Mask R-CNN (He et al., 2017) adopt the two-stage object detector. Besides, their ROI pooling or aligning strategy would lose details of the cells. Moreover, they generally ignore the rich low-level feature on the shallow layers. In this paper, we elegantly combine the one-stage SSD detector with U-net, making the neural cell instance segmentation more accurate and fast.

4.5.2. Qualitative results

The qualitative results of neural cell instance segmentation are displayed in Fig. 9. We observe that the unsupervised method IMDLMS (Peng et al., 2009) can hardly capture the boundaries of the neural cells. To improve its accuracy, we have to adjust its parameters exhaustively for each image. However, it costs enormous time when applying the method to a bunch of images.

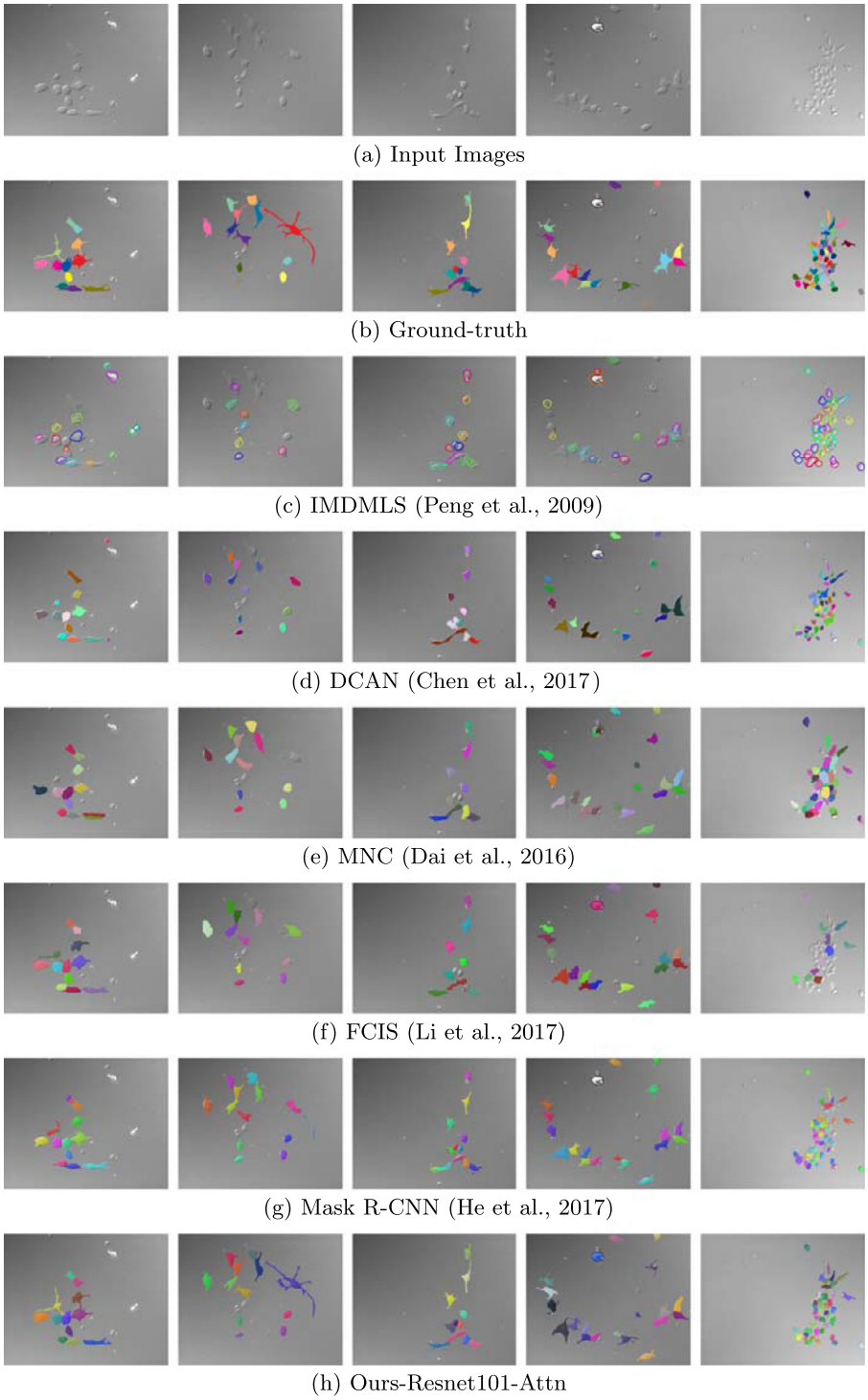


Fig. 9. Qualitative neural cell instance segmentation results. Compared with the other methods, Ours-ResNet101-Attn is more accurate especially in capturing the slender and tiny structures of neural cells.

DCAN (Chen et al., 2017) can hardly capture the slender structures of the neural cells due to the discarding of the contours. Besides, DCAN tends to make severe over-segmentation when cells are too close to each other. MNC (Dai et al., 2016) uses fully connected layer (FC) for semantic segmentation, while Mask R-CNN (He et al., 2017) adopts CNNs for semantic segmentation. The results in Fig. 9 show that the CNN layers capture more details than the FC layers. However, the CNNs consume much more time (see Table 4). FCIS (Li et al., 2017) adopts pixel-wise softmax to obtain the instance mask, although fast, it is prone to generate the wavy artifacts in the cell boundaries. Besides, FCIS is weak in detecting small cells. Mask R-CNN is better in predicting the tiny structures of small cells due to its FPN detector. However, when the cell instance is large, the Mask R-CNN also fails to capture the slender structure as it ignores the high-resolution details in the shallow layers. Moreover, Mask R-CNN adopts the ROI align to restrict the object feature patch to 7×7. The low resolution is not enough to describe the details, especially for large objects. Compared to

these state-of-the-art methods, our model exhibit remarkable performance in capturing the tiny and slender structures of neural cells. The results indicate that the shallow feature maps are essential in capturing the detailed pixel-wise information.

4.5.3. Segmentation attention unit

To make a further exploration of the attention unit effect, we compared the qualitative results of the model with and without segmentation attention unit (Fig. 10). There are multiple cases in our prediction that the bounding box of one cell would involve the other parts of another cell. The U-net dramatically suppresses the false predictions by gradually introducing semantics from deep layers to shallow layers. However, we observe that when a big cell bounding box completely involves another small cell, such as the case in Fig. 10, there still are some false pixel-wise segmentation (pointed by red arrows). We conjecture that the high percent involvement of the bounding boxes make network think the false segmentation part belongs to the big target. Besides, the lack of such cell images makes the situation worse since the network does not have experience in handling these cases. The introduction of the attention unit enhances the useful features, therefore suppressing the false predictions (Fig. 10(e)).

4.6. Visualization of the attention map

The attention map (Eq. (6)) at different query points from conv3 to conv6 are visualized in Fig. 11. The attention map describes where the model pays attention to when a specific location on the feature map is queried. Note that the sizes of the feature maps are decreasing from conv3 to conv6, and the solid circles represent the projected regions on the input images. From Fig. 11 we can observe that the model would consider other similar cell regions when part of the cells are queried. Besides, the model will consult all the background pixels when a background position is queried. On the deepest layer conv6, the model only attends to the largest cell. Fig. 11 also reveals that attention is mainly applied to the soma of the cells, suggesting that our model can be used for non-neuronal cells.

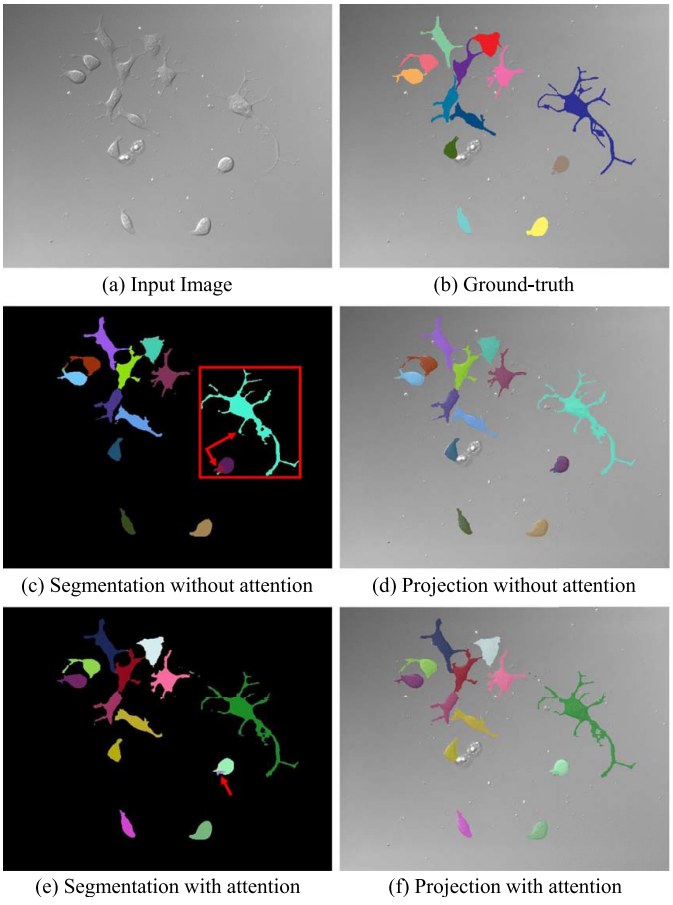


Fig. 10. Effect of the segmentation attention unit. (c) and (e) show the segmentation results. (d) and (f) are the segmentation results projected to the input image. The red box in (c) indicates the bounding box of the green cell. The red arrows point to the green cell and the false segmentation that belongs to the green cell. The red arrow in (e) shows that the segmentation attention unit suppresses false segmentation.

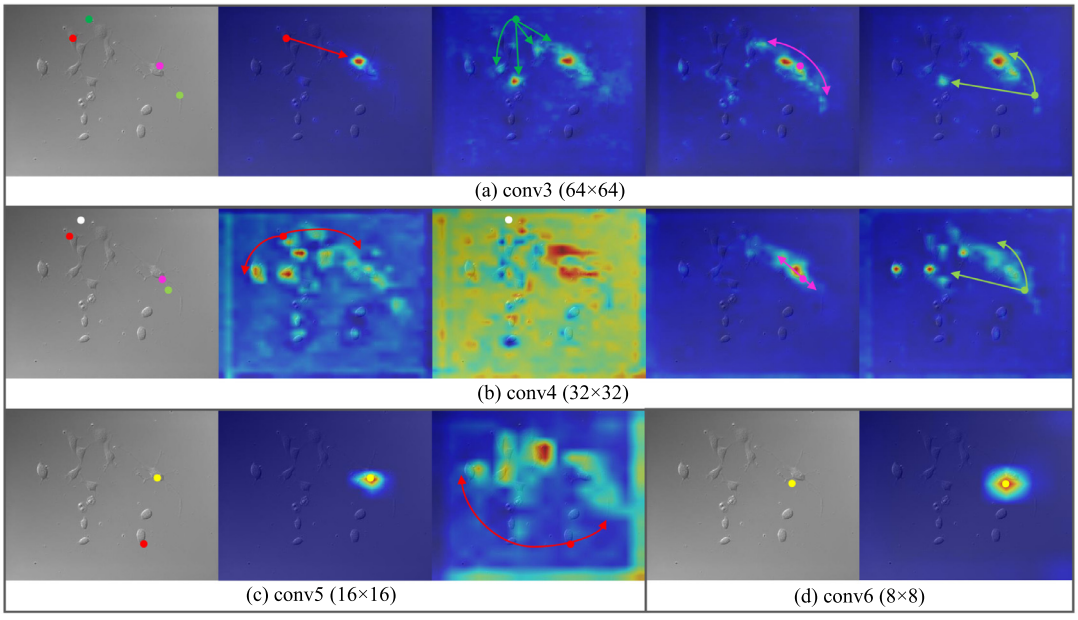


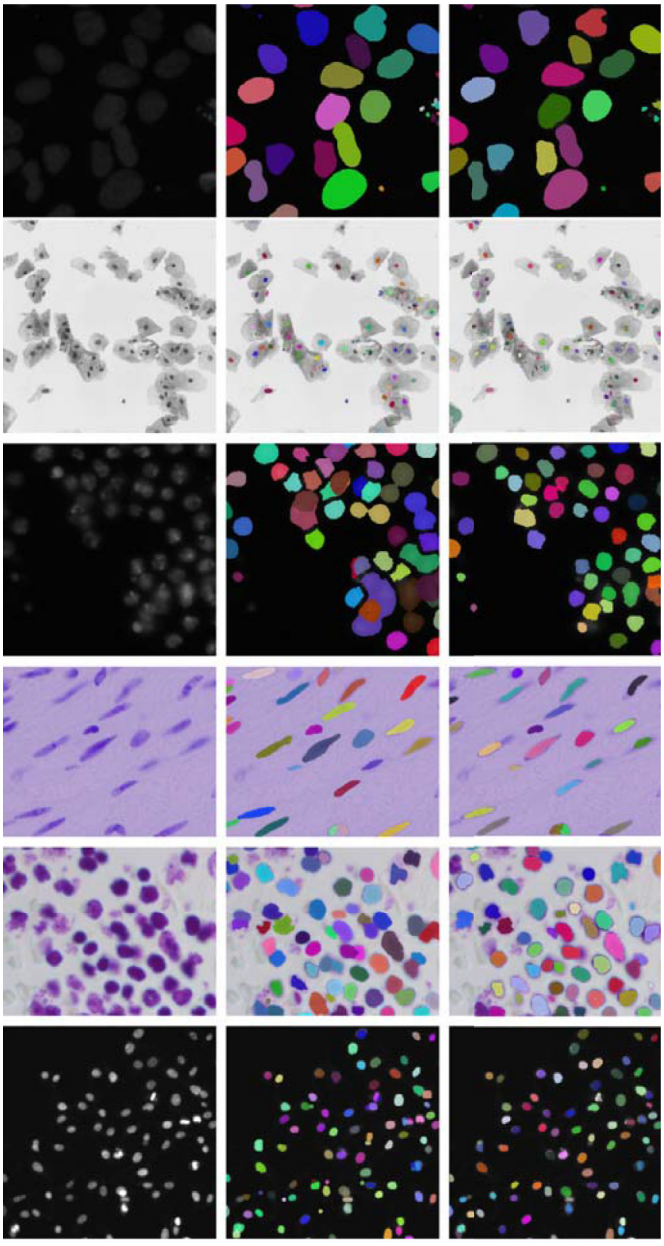
Fig. 11. Visualization of the attention maps at different query points. The query points are represented by solid circles. Note that the locations of the query points are on the feature maps, whose sizes vary across different convolutional layers. The leftmost image in each sub-figure is the input image, while other images are the attention maps that are resized and projected on the input image. The response of a position centered at the solid circle is computed by the weighted average of the attention features of all positions.

Table 5
Quantitative results of nucleus instance segmentation of our method on the Kaggle 2018 Data Science Bowl dataset. The architecture is under the same settings as neural cells.

Method	AP@0.5	AP@0.7	AIoU@0.5	AIoU@0.7
CosineEmbedding (Payer et al., 2018)	17.87	3.41	64.14	76.84
DCAN (Chen et al., 2017)	51.88	23.45	74.08	82.56
Mask R-CNN (He et al., 2017)	69.88	54.69	80.57	**84.83**
Ours	**70.93**	**56.88**	**80.88**	84.55

4.7. Application to other datasets

To investigate the generalization ability of our method, we apply our method to a public dataset of the Kaggle competition: 2018 Data Science Bowl. The aim of the competition is to find the nuclei in divergent images, which are acquired under a variety of conditions and vary in the cell type, magnification, and imaging modality (brightfield vs. fluorescence). The dataset does not contain neurons or cell types with filopodia-like structures. It is adopted as an extended experiment to show that our method could be applied to different types of images.



(a) Input images (b) Ground-truth (c) Ours

Fig. 12. Qualitative results of nucleus instance segmentation of our method on a public dataset of Kaggle 2018 Data Science Bowl. The dataset images are acquired under a variety of conditions and vary in the cell type, magnification, and imaging modality (brightfield vs. fluorescence).

In particular, we separate the original training datasets (670 images) with annotations into three sets: training (402 images), validation (134 images), testing (134 images). These images vary in sizes. We use the same architecture settings, data augmentations, implementation skills and evaluation metrics as the neural cells. Note that the backbone of the network is initialized with the training weights from ImageNet, other weights are sampled from a Gaussian Distribution. We divide the input images by 255 to normalize the images before training, validation and testing. The quantitative and qualitative results on the testing set are reported in Table 5 and Fig. 12, respectively. The results demonstrate that our method can be applied to other biomedical instance segmentation tasks.

Although the cell nuclei are in regular morphology, the Kaggle Science Bowl dataset is challenging. First, the cell images are acquired under different conditions. Second, the cell nuclei's morphologies are quite different for each type. Third, the number of images for each type or each imaging condition is largely imbalanced. Finally, there are some human annotation errors in the dataset. Compared to the neural cell dataset (386 images for training), the Kaggle dataset does not provide sufficient data for the model to fully learn all types of cell nuclei under different imaging conditions (402 images for training in total). These are the reason why the performance of our method on the Kaggle dataset (Table 5) is not as good as the performance on neural cell dataset (Table 4). However, it is still better than the other state-of-the-art methods. In particular, CosineEmbedding (Payer et al., 2018) largely suffers from the false positive detections, which are caused by the undesired behavior that the clustering usually generates multiple separate clusters for the same cell instances. DCAN (Chen et al., 2017) is less effective at separating the touching cells due to the unclear cell boundaries. Mask R-CNN (He et al., 2017) achieves similar performance to our method because bounding boxes are superior in separating the touching cells. Nevertheless, overall our method is more effective than Mask R-CNN.

5. Conclusion

This paper proposes an attentive neural cell instance segmentation method. The employment of the one-stage object detector makes the proposed model accurate and fast in detection. Besides, the proposed attention modules enhance the model ability to learn the cell objectness information as well as capture the slender and tiny structures of the cells. These properties indicate a great potential of our method in physical interaction study between neural stem cells.

Future endeavors will be devoted to the expansion of performance testing on young neurons. Besides, we plan to investigate the effect of the proposed model on other biomedical images that depict tiny and slender objects, such as vessel images.

Conflict of interest

This manuscript is an extension of our previous works:

1. Pixel-wise neural cell instance segmentation. ISBI, 2018.
2. Instance segmentation of neural cells. ECCV Workshop, 2018.

The manuscript inherits the main instance segmentation network that jointly employs SSD and U-net. The manuscript additionally proposes two kinds of attention mechanism to improve the accuracy of both detection and segmentation. Moreover, the manuscript adds more ablations studies, comparison experiments, mathematical formulations and extended application.

References

Al-Kofahi, Y., Lassoued, W., Lee, W., Roysam, B., 2010. Improved automatic detection and segmentation of cell nuclei in histopathology images. IEEE Trans. Biomed. Eng. 57 (4), 841–852.

Althoff, K., Degerman, J., Gustavsson, T., 2005. Combined segmentation and tracking of neural stem-cells. In: Proc. Scand. Conf. Image Anal., pp. 282–291.

Bensch, R., Ronneberger, O., 2015. Cell segmentation and tracking in phase contrast images using graph cut with asymmetric boundary costs. In: Proc. IEEE Int. Symp. Biomed. Imaging, pp. 1220–1223.

Bernardis, E., Yu, S.X., 2010. Finding dots: segmentation as popping out regions from boundaries. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 199–206.

Boykov, Y.Y., Jolly, M., 2001. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In: Proc. IEEE Int. Conf. Comput. Vis., pp. 105–112.

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.

Chen, H., Qi, X., Yu, L., Dou, Q., Qin, J., Heng, P.-A., 2017. Dcan: deep contour-aware networks for object instance segmentation from histology images. Med. Image Anal. 36, 135–146.

Chen, L.-C., Hermans, A., Papandreou, G., Schroff, F., Wang, P., Adam, H., 2018. MaskLab: instance segmentation by refining object detection with semantic and direction features. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 4013–4022.

Chen, L.-C., Yang, Y., Wang, J., Xu, W., Yuille, A.L., 2016. Attention to scale: scale-aware semantic image segmentation. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 3640–3649.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20 (3), 273–297.

Dai, J., He, K., Sun, J., 2016. Instance-aware semantic segmentation via multi-task network cascades. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 3150–3158.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 886–893.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Li, F.-F., 2009. ImageNet: a large-scale hierarchical image database. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 248–255.

Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A., 2010. The PASCAL visual object classes (VOC) challenge. Int. J. Comput. Vis. 88 (2), 303–338.

Freund, Y., Schapire, R.E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. 55 (1), 119–139.

Fu, C.-Y., Liu, W., Ranga, A., Tyagi, A., Berg, A. C., 2017. DSSD: deconvolutional single shot detector. arXiv:1701.06659.

Garcia, E., Hermoza, R., Castanon, C.B., Cano, L., Castillo, M., Castanñeda, C., 2017. Automatic lymphocyte detection on gastric cancer IHCimages using deep learning. In: Proc. IEEE Int. Symp. on Computer-Based Medi. Sys., pp. 200–204.

Girshick, R.B., 2015. Fast R-CNN. In: Proc. IEEE Int. Conf. Comput. Vis., pp. 1440–1448.

Girshick, R.B., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 580–587.

He, K., Gkioxari, G., Dollár, P., Girshick, R.B., 2017. Mask R-CNN. In: Proc. IEEE Int. Conf. Comput. Vis., pp. 2980–2988.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 770–778.

Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y., 2018. Relation networks for object detection. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 3588–3597.

Ioffe, S., Szegedy, C., 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proc. Int. Conf. Mach. Learn., pp. 448–456.

Kainz, P., Urschler, M., Schulter, S., Wohlhart, P., Lepetit, V., 2015. You should use regression to detect cells. In: Proc. Med. Image Comput. Comput. Assist. Interv., Part 3, pp. 276–283.

Kass, M., Witkin, A., Terzopoulos, D., 1988. Snakes: active contour models. Int. J. Comput. Vis. 1 (4), 321–331.

Koyuncu, C.F., Arslan, S., Durmaz, I., Cetin-Atalay, R., Gunduz-Demir, C., 2012. Smart markers for watershed-based cell segmentation. PloS one 7 (11), e48664.

Li, C., Wang, X., Liu, W., Latecki, L.J., 2018. Deepmitosis: mitosis detection via deep detection, verification and segmentation networks. Med. Image Anal. 45, 121–133.

Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y., 2017. Fully convolutional instance-aware semantic segmentation. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 4438–4446.

Lienhart, R., Maydt, J., 2002. An extended set of Haar-like features for rapid object detection. In: Proc. IEEE Int. Conf. Image Process., pp. 900–903.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C., Berg, A.C., 2016. SSD: single shot multibox detector. In: Proc. Eur. Conf. Comput. Vis., Part 1, pp. 21–37.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 3431–3440.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60 (2), 91–110.

López-Linares, K., Aranjuelo, N., Kabongo, L., Maclair, G., Lete, N., Ceresa, M., García-Familiar, A., Macía, I., Ballester, M.A.G., 2018. Fully automatic detection and segmentation of abdominal aortic thrombus in post-operative cta images using deep convolutional neural networks. Med. Image Anal. 46, 202–214.

Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted Boltzmann machines. In: Proc. Int. Conf. Mach. Learn., pp. 807–814.

Noh, H., Hong, S., Han, B., 2015. Learning deconvolution network for semantic segmentation. In: Proc. IEEE Int. Conf. Comput. Vis., pp. 1520–1528.

Otsu, N., 1979. A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. 9 (1), 62–66.

Payer, C., Štern, D., Neff, T., Bischof, H., Urschler, M., 2018. Instance segmentation and tracking with cosine embeddings and recurrent hourglass networks. In: Frangi, A.F., Schnabel, J.A., Davatzikos, C., Alberola-López, C., Fichtinger, G. (Eds.), Med. Image Comput. Comput. Assist. Interv.. Springer International Publishing, Cham, pp. 3–11.

Peng, H., Zhou, X., Li, F., Xia, X., Wong, S.T.C., 2009. Integrating multi-scale blob/curvilinear detector techniques and multi-level sets for automated segmentation of stem cell images. In: Proc. IEEE Int. Symp. Biomed. Imaging, pp. 1362–1365.

Pinidiyaarachchi, A., Wählby, C., 2005. Seeded watersheds for combined segmentation and tracking of cells. In: Proc. Int. Conf. Image Anal. Process., pp. 336–343.

Ravin, R., Hoeppner, D.J., Munno, D.M., Carmel, L., Sullivan, J., Levitt, D.L., Miller, J.L., Athaide, C., Panchision, D.M., McKay, R.D.G., 2008. Potency and fate specification in CNS stem cell populations in vitro. Cell Stem Cell 3 (6), 670–680.

Redmon, J., Divvala, S., Girshick, R.B., Farhadi, A., 2016. You only look once: unified, real-time object detection. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 779–788.

Ren, S., He, K., Girshick, R.B., Sun, J., 2015. Faster R-CNN: towards real-time object detection with region proposal networks. In: Proc. Neural Inf. Process. Syst., pp. 91–99.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-Net: convolutional networks for biomedical image segmentation. In: Proc. Med. Image Comput. Comput. Assist. Interv., Part 3, pp. 234–241.

Sahoo, P.K., Soltani, S., Wong, A.K.C., Chen, Y.C., 1988. A survey of thresholding techniques. Comput. Vis. Graph. Image Process. 41 (2), 233–260.

Sankaran, P., Asari, V.K., 2006. Adaptive thresholding based cell segmentation for cell-destruction activity verification. In: Proc. IEEE Applied Imagery and Pattern Recognition Workshop. 14–14

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: Proc. Int. Conf. Learn. Represent..

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Proc. Neural Inf. Process. Syst., pp. 6000–6010.

Vincent, L., 1993. Morphological grayscale reconstruction in image analysis: applications and efficient algorithms. IEEE Trans. Image Process. 2 (2), 176–201.

Vincent, L., Soille, P., 1991. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans. Pattern Anal. Mach. Intell. 13 (6), 583–598.

Vink, J.P., van Leeuwen, M.B., van Deurzen, C.H.M., de Haan, G., 2013. Efficient nucleus detector in histopathology images. J. Microsc. 249 (2), 124–135.

Wang, M., Zhou, X., Li, F., Huckins, J., King, R.W., Wong, S.T.C., 2008. Novel cell segmentation and online SVM for cell cycle phase identification in automated microscopy. Bioinformatics 24 (1), 94–101.

Wang, X., Girshick, R., Gupta, A., He, K., 2018. Non-local neural networks. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 7794–7803.

Wu, P., Yi, J., Zhao, G., Huang, Z., Qiu, B., Gao, D., 2015. Active contour-based cell segmentation during freezing and its application in cryopreservation. IEEE Trans. Biomed. Eng. 62 (1), 284–295.

Xie, S., Girshick, R.B., Dollár, P., Tu, Z., He, K., 2017. Aggregated residual transformations for deep neural networks. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 5987–5995.

Yang, D., Huang, Q., Axel, L., Metaxas, D., 2018. Multi-component deformable models coupled with 2d-3d u-net for automated probabilistic segmentation of cardiac walls and blood. In: Proc. IEEE Int. Symp. Biomed. Imaging, pp. 479–483.

Yang, L., Tuzel, O., Meer, P., Foran, D.J., 2008. Automatic image analysis of histopathology specimens using concave vertex graph. In: Proc. Med. Image Comput. Comput. Assist. Interv., Part 1, pp. 833–841.

Yi, J., Wu, P., Hoeppner, D.J., Metaxas, D.N., 2017. Fast neural cell detection using light-weight SSD neural network. In: Proc. IEEE Comput. Vis. Pattern Recognit. Workshop, pp. 860–864.

Yi, J., Wu, P., Hoeppner, D.J., Metaxas, D.N., 2018a. Pixel-wise neural cell instance segmentation. In: Proc. IEEE Int. Symp. Biomed. Imaging, pp. 373–377.

Yi, J., Wu, P., Jiang, M., Hoeppner, D.J., Metaxas, D.N., 2018b. Instance segmentation of neural cells. In: Proc. Eur. Conf. Comput. Vis. Workshop.

Zhang, C., Yarkony, J., Hamprecht, F.A., 2014. Cell detection and segmentation using correlation clustering. In: Proc. Med. Image Comput. Comput. Assist. Interv., Part 1, pp. 9–16.

Zhang, H., Dana, K., Shi, J., Zhang, Z., Wang, X., Tyagi, A., Agrawal, A., 2018a. Context encoding for semantic segmentation. In: Proc. IEEE Comput. Vis. Pattern Recognit., pp. 7151–7160.

Zhang, H., Goodfellow, I., Metaxas, D., Odena, A., 2018b. Self-attention generative adversarial networks. arXiv preprint arXiv:1805.08318.