NISNet3D: Three-Dimensional Nuclear Synthesis and Instance Segmentation for Fluorescence Microscopy Images

Liming Wu[1], Alain Chen[1], Paul Salama[2], Kenneth Dunn[3] and Edward Delp[1]

[1]Video and Image Processing Laboratory, School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana 47907, United States.
[2]Department of Electrical and Computer Engineering, Indiana University-Purdue University Indianapolis, Indianapolis, Indiana 46202, United States.
[3]School of Medicine, Indiana University, Indianapolis, Indiana 46202, United States.

Contributing authors: wu1114@purdue.edu; chen2967@purdue.edu; psalama@iupui.edu; kwdunn@iu.edu; ace@ecn.purdue.edu;

Abstract

The primary step in tissue cytometry is the automated distinction of individual cells (segmentation). Since cell borders are seldom labeled, researchers generally segment cells by their nuclei. While effective tools have been developed for segmenting nuclei in two dimensions, segmentation of nuclei in three-dimensional volumes remains a challenging task for which few tools have been developed. The lack of effective methods for three-dimensional segmentation represents a bottleneck in the realization of the potential of tissue cytometry, particularly as methods of tissue clearing present researchers with the opportunity to characterize entire organs. Methods based upon deep-learning have shown enormous promise, but their implementation is hampered by the need for large amounts of manually annotated training data. In this paper we describe 3D Nuclei Instance Segmentation

Network (NISNet3D), a deep learning-based approach in which training is accomplished using synthetic data, profoundly reducing the effort required for network training. We compare results obtained from NISNet3D with results obtained from eight existing techniques.

Keywords: Fluorescence microscopy images, generative adversarial networks, nuclear instance segmentation, synthetic microscopy image generation

1 Introduction

Over the past ten years, various technological developments have provided biologists with the ability to collect microscopy images of enormous scale and complexity. Methods of tissue clearing combined with automated confocal or lightsheet microscopes have enabled three-dimensional imaging of entire organs or even entire organisms at subcellular resolution. Novel methods of multiplexing have been developed so that researchers can now simultaneously characterize 50 or more targets in the same tissue. However, as biologists turn to the task of analyzing these extraordinary volumes (tissue cytometry), they quickly discover that the methods of automated image analysis necessary for extracting quantitative data from images of this scale are frequently inadequate for the task. In particular, while effective methods for distinguishing (segmenting) individual cells are available for analyses of two-dimensional images, corresponding methods for segmenting cells in three-dimensional volumes are generally lacking. The problem of three-dimensional image segmentation thus represents a bottleneck in the full realization of 3D tissue cytometry as a tool in biological microscopy.

There are two main categories of segmentation approaches, non-machine-learning image processing and computer vision techniques and techniques based on machine learning and in particular deep learning [1, 2]. The traditional image processing techniques (e.g. watershed, thresholding, edge detection, and morphological operations) can be effective on one type of images but may not generalize to other types of images without careful parameter tuning. Segmentation techniques based on deep learning have shown great promise, in some cases providing accurate and robust results across a range of image types [3–7]. However, their utility is limited by the large amounts of manually annotated (ground truth) data needed for training, validation, and testing. Annotation is a labor-intensive and time-consuming process, especially for a 3D volume. While tools have been developed to facilitate the laborious process of manual annotation [8–11], the generation of training data for microscopy images remains a major obstacle to implementing segmentation approaches based upon deep learning.

The problem of generating sufficient training data can be alleviated using data augmentation, a process in which existing manually annotated training data is supplemented with synthetic data generated from modifications of the

manually annotated data [12–15]. An alternative approach is to use synthetic data for training [14, 16, 17]. A method described in [18] generates synthetic 3D microscopy volumes by stacking 2D synthetic image slices using 2D distributions of fluorescent markers that are generated using Generative Adversarial Networks (GANs). Similarly, a 3D GAN was used in [19] to generate fully 3D volumetric cell masks. We have shown [7, 20–22] that GANs can be used to generate synthetic microscopy volumes that can be used for training, and incorporated this approach into the DeepSynth segmentation system [6].

Convolutional Neural Networks (CNNs) have had great success for solving problems such as object classification, detection, and segmentation [23, 24]. The encoder-decoder architecture has been widely used for biomedical image analysis including volumetric segmentation [25–27], medical image registration [28], and nuclear segmentation [3, 6, 15, 29–33]. However, most CNNs are designed for segmenting two-dimensional images and cannot be directly used for segmentation of 3D volumes [29, 32, 34]. Other methods described in [3, 14, 30] process images slice by slice and fuse together two dimensional results to form 3D segmentation results that fail to represent the 3D anisotropy of microscopy volumes.

In this paper we describe the 3D Nuclei Instance Segmentation Network (NISNet3D), a deep learning-based segmentation technique that can use synthetic volumes, manually annotated volumes, or a combination of synthetic and annotated volumes. NISNet3D is a true three-dimensional segmentation method that operates directly on 3D volumes, using 3D CNNs to exploit 3D information in a microscopy volume, thereby generating more accurate segmentations of nuclei in 3D image volumes.

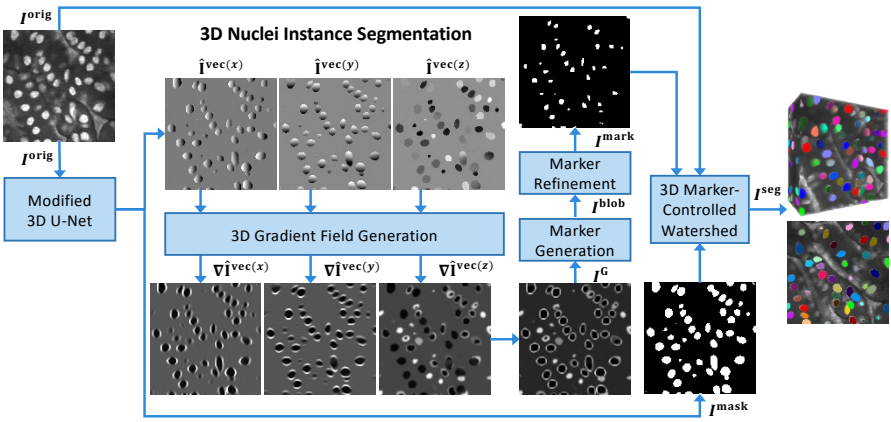

Fig. 1 Overview of NISNet3D for 3D nuclei instance segmentation. NISNet3D uses a modified 3D U-Net for nuclei segmentation and 3D vector field array estimation where each voxel represents a 3D vector pointing to the nearest nuclei centroid. NISNet3D then generates a 3D gradient field array from the 3D vector field array and further generates refined markers for 3D watershed segmentation

2 Results

2.1 3D Nuclei Image Synthesis

Deep learning methods generally require large amounts of training samples to achieve accurate results. However, manually annotating ground truth is a tedious task and impractical in many situations especially for 3D microscopy volumes. To address this issue, we generate synthetic microscopy volumes for training NISNet3D. In particular, we enhance the synthetic microscopy volume generation method described in [6, 7, 31] to model non-ellipsoidal or irregularly shaped nuclei, in contrast to the prior methods described in [6, 7] which model nuclei as regular ellipsoids using the approach described in [31]. It must be emphasized that NISNet3D can use synthetic volumes or annotated real volumes or combinations of both. We will demonstrate this in our experiments.

For generating microscopy volumes, we first generate synthetic segmentation masks, which will be used as the ground truth masks during training, by iteratively adding initial binary nuclei to an empty 3D volume. The details for generating initial binary nuclei are described in Section 3.2.1. Since many of the nuclei in the original microscopy images are not strictly ellipsoidal and look more like deformed ellipsoids (See Figure 2 (left)), We model these initial nuclei as deformed 3D binary ellipsoids with random size and orientation. Specifically, we use elastic transformation [35] to deform the 3D binary mask of the nuclei. The details for deforming the initial binary nuclei are described in Section 3.2.1.

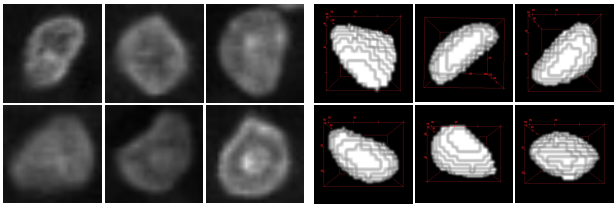

Fig. 2 Non-ellipsoidal shaped nuclei in actual microscopy volumes (left column) and synthetic binary nuclei segmentation masks after using elastic deformation (right column)

Examples of deformed ellipsoids are shown in Figure 2 (right column). We then use the unpaired image-to-image translation model known as SpCycleGAN [7, 36] for generating synthetic microscopy volumes. By unpaired we mean that the binary segmentation mask we created above is not the ground truth of actual microscopy images. The input to the SpCycleGAN is the binary segmentation masks we created and actual microscopy images (i.e. unpaired). As shown in Figure 1, we use the binary segmentation mask we created and actual microscopy volumes for training the SpCycleGAN. After training we generate synthetic microscopy volumes by using different synthetic microscopy segmentation masks we created as input to the SpCycleGAN. Note that since the SpCycleGAN generates 2D slices, we sequentially generate 2D slices for

each focal plane and stack them to construct a 3D synthetic volume. During our experiments, we observed that if we only model nuclei as regular ellipsoids when the original microscopy volume contains many non-ellipsoidal or irregularly shaped nuclei, the SpCycleGAN will have difficulty learning the mapping function between the the binary masks and microscopy images, resulting in generating many non-ellipsoidal nuclei which are not aligned well with the corresponding binary masks. Including non-ellipsoidal nuclei or irregularly shaped nuclei will help SpCycleGAN learn a more accurate mapping and thus the SpCycleGAN can generate more representative synthetic images. We have generated 950 synthetic microscopy volumes in total, and each is of size $128 \times 128 \times 128$ using the representative subvolumes of different datasets. The examples of actual microscopy images and corresponding synthetic microscopy images are shown in Figure 3.

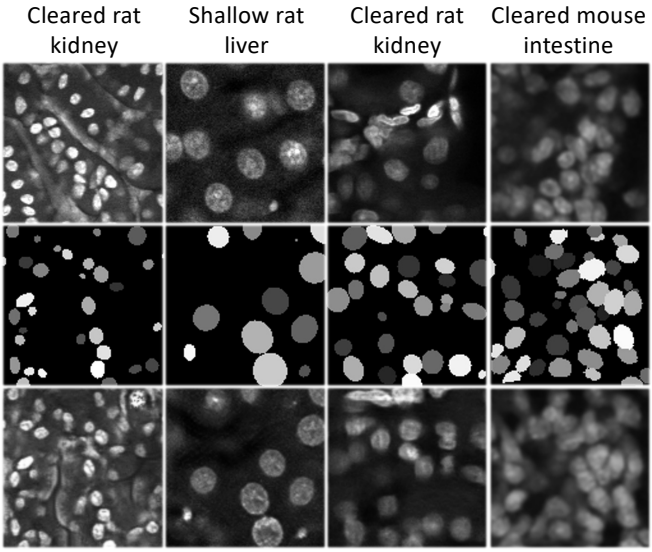

Fig. 3 Actual microscopy images (top row), synthetic segmentation masks (middle row), and corresponding synthetic microscopy images (bottom row)

2.2 Evaluation Datasets

We used four microscopy volumes in our experiments, denoted by $\mathcal{V}_1$-$\mathcal{V}_4$, having fluorescent-labeled (Hoechst 33342 stain) nuclei that were collected from cleared rat kidneys, shallow rat livers, and cleared mouse intestines using confocal microscopy, to both generate synthetic data as well as provide ground truth data. Datasets $\mathcal{V}_1$-$\mathcal{V}_4$ are available at [37]. In particular, we use the 3D Nuclei Image Synthesis method described in Section 2.1 in conjunction with a subset of $\mathcal{V}_1$-$\mathcal{V}_4$ (other than those used for validation and testing) to generate 950 volumes of corresponding synthetic microscopy volumes (250 volumes for $\mathcal{V}_1$, 250 volumes for $\mathcal{V}_2$, 250 volumes for $\mathcal{V}_3$, and 200 volumes for $\mathcal{V}_4$), that

Table 1 The description of the five datasets used for evaluation

Original Microscopy Volumes			
Volume ID	**Volume Description**	**Original Size ($X \times Y \times Z$)**	**Annotated Subvolume Size**
$\mathcal{V}_1$	cleared rat kidney	$512 \times 512 \times 200$	$128 \times 128 \times 64$
$\mathcal{V}_2$	shallow rat liver	$512 \times 512 \times 32$	$128 \times 128 \times 32$
$\mathcal{V}_3$	cleared rat kidney	$512 \times 512 \times 415$	$128 \times 128 \times 64$
$\mathcal{V}_4$	cleared mouse intestine	$512 \times 930 \times 157$	$128 \times 128 \times 40$
$\mathcal{V}_5$	zebrafish brain	$2000 \times 1450 \times 397$	$64 \times 64 \times 64$

were then used for training NISNet3D. We also manually annotated representative subvolumes of $\mathcal{V}_1$-$\mathcal{V}_4$ using ITK-SNAP [8], which are used as ground truth data. We have annotated 1, 16, 9, 5 subvolumes for $\mathcal{V}_1$-$\mathcal{V}_4$, respectively. It is to be noted that both $\mathcal{V}_1$ and $\mathcal{V}_3$ are cleared rat kidney volumes but show different nuclear morphology. In addition, we also used a publicly available electron microscopy zebrafish brain volume, $\mathcal{V}_5$, known as NucMM [38], to evaluate NISNet3D. We use the NucMM dataset to demonstrate that NISNet3D not only works on fluorescent microscopy volumes but also works with other high resolution imaging modalities. Detailed information of all original five datasets used in our evaluation is shown in Table 1.

2.3 NISNet3D

Nuclei instance segmentation methods typically include a foreground and background separation step and a nuclei instance identification and segmentation step. Non-machine learning methods use traditional thresholding such as Otsu's thresholding for separating background and foreground and use a watershed transformation to identify individual cell nuclei. However, classical thresholding methods are sensitive to variability in intensity and may not generate accurate segmentation masks. Also, watershed segmentation without accurate markers may generate under-segmentation or over-segmentation results. Machine learning methods typically generate more accurate results with enough training data. In addition, most current methods that are designed for nuclei instance segmentation only works on 2D images or uses a 2D to 3D reconstruction, which may not fully capture the 3D spatial information.

We improved upon these methods by designing a nuclei segmentation network, NISNet3D, using a modified 3D U-Net with attention and residual blocks, which can directly generate volumetric segmentation results. The modified 3D U-Net can be trained on both synthetic microscopy volumes and actual microscopy volumes if manual ground truth annotations are available. We generate more accurate markers than the direct distance transform used in classic watershed by learning a 3D vector field array which contains information that can be used to accurately infer the location of nuclei centroids as well nuclei boundaries. As shown in Figure 1, using the 3D vector field array, we further generate a 3D gradient array which contains gradient information,

to aid in the detection of nuclei boundary and which also enables the generation and refinement of 3D markers that are to be used by a watershed method to perform segmentation. In addition, we provide two versions of NISNet3D: NISNet3D-slim and NISNet3D-syn that have the same network architectures but are trained with different datasets. NISNet3D-slim was trained only on a small number of synthetic/original volumes whereas NISNet3D-synth was trained using all the synthetic datasets we have. The details of NISNet3D-synth and NISNet3D-slim training and evaluation are described in Section 3.5. As demonstrated in Section 2.4 our methods achieve more accurate segmentation results.

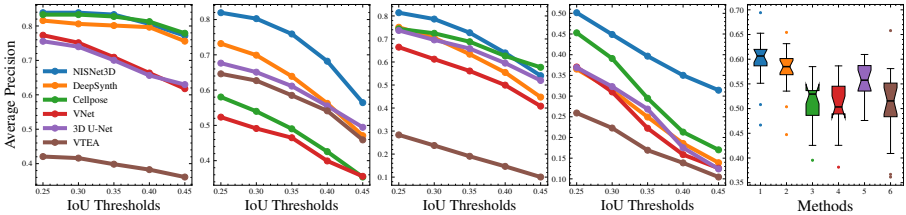

Fig. 4 Evaluation results using Average Precision (AP) for multiple Intersection-over-Unions (IoUs) thresholds, T_{IoUs}, for datasets $\mathcal{V}_1$-$\mathcal{V}_4$, and box plots of Aggregated Jaccard Index (AJI) of each subvolume in dataset $\mathcal{V}_2$

2.4 Evaluation

We compared the NISNet3D with deep learning image segmentation methods including VNet [27], 3D U-Net [26], Cellpose [3], DeepSynth [6], and StarDist3D [15]. In addition, we also compare NISNet3D with several commonly used biomedical image processing tools including 3D Watershed [39], Squassh [40], CellProfiler [41], and VTEA [42]. We trained and evaluated the comparison methods using the same dataset as used for NISNet3D. We also used the same training and evaluation strategies as NISNet3D described in Section 3.5.

We use object-based metrics to evaluate nuclei instance segmentation accuracy. To reduce the bias [43], we use the mean Precision, mean Recall and mean F_1 score on multiple IoU thresholds. We set $T_{\text{IoUs}} = \{0.25, 0.3, ..., 0.45\}$ for datasets $\mathcal{V}_1$-$\mathcal{V}_4$, and set $T_{\text{IoUs}} = \{0.5, 0.55, ..., 0.75\}$ for datset $\mathcal{V}_5$. The selection of the T_{IoUs} is described in more detail below. We observed that the nuclei in datasets $\mathcal{V}_1$-$\mathcal{V}_4$ are more challenging to segment than the nuclei in $\mathcal{V}_5$. If we use the same IoU thresholds for evaluating all the datasets, the evaluation accuracy for $\mathcal{V}_1$-$\mathcal{V}_4$ will be much lower than the evaluation accuracy for $\mathcal{V}_5$ for all compared methods. Thus, we chose two different sets of IoU thresholds for $\mathcal{V}_1$-$\mathcal{V}_4$ and $\mathcal{V}_5$, respectively.

We also examined commonly used object detection metrics: Average Precision (AP) [44, 45] by estimating the area under the Precision-Recall Curve [46] using the same thresholds for T_{IoUs} as described in the previous paragraph.

Table 2 Comparison of object-based evaluation metrics for microscopy datasets $\mathcal{V}_1$ - $\mathcal{V}_4$. The best performance with respect to each metric are in bold

Methods	Microscopy $\mathcal{V}_1$				Microscopy $\mathcal{V}_2$				Microscopy $\mathcal{V}_3$				Microscopy $\mathcal{V}_4$			
	mP	mR	mF_1	mAP	mP	mR	mF_1	mAP	mP	mR	mF_1	mAP	mP	mR	mF_1	mAP
3D Watershed [39]	73.77	63.60	68.31	48.87	70.13	83.80	76.36	60.30	78.43	58.40	66.95	46.73	27.51	33.66	30.22	11.96
Squassh [40]	61.49	14.56	23.54	9.21	84.08	73.15	78.24	61.80	65.48	16.53	26.40	12.26	2.96	11.95	4.73	0.49
CellProfiler [41]	61.67	57.31	59.41	37.79	69.12	82.20	75.09	57.52	59.18	42.46	49.44	26.87	27.77	27.86	27.75	10.15
VTEA [42]	68.91	55.76	61.64	42.71	74.44	76.73	75.57	57.76	43.36	51.56	47.11	23.00	30.23	52.90	38.45	17.89
VNet [27]	91.47	74.98	82.41	70.29	80.52	53.95	64.61	45.88	79.85	70.04	74.56	58.33	36.12	50.26	41.73	23.74
3D U-Net [26]	**92.12**	73.57	81.81	69.63	**86.08**	65.85	74.62	58.12	**85.40**	73.32	78.89	64.21	38.82	51.43	43.93	25.19
Cellpose [3]	89.47	89.47	89.47	81.70	66.07	67.79	66.66	47.80	81.98	78.58	80.24	67.19	**58.56**	45.01	50.67	30.43
DeepSynth [6]	81.09	**94.28**	87.19	79.51	71.86	**88.75**	79.42	66.19	83.29	73.14	77.85	62.86	45.45	47.37	46.34	23.57
NISNet3D-slim	88.69	91.21	**89.93**	**81.97**	81.25	87.95	**84.47**	**72.16**	82.73	**83.31**	**83.00**	**71.18**	56.40	**60.69**	**58.43**	**37.30**
Cellpose-synth [3]	74.26	**92.31**	82.31	72.35	**73.64**	86.57	79.57	67.59	**69.40**	72.69	70.98	53.62	53.24	43.53	47.80	26.39
StarDist3D-synth [15]	85.77	74.67	79.83	66.81	63.78	57.54	60.47	40.00	74.73	48.12	58.40	39.52	**74.73**	48.12	58.40	39.52
NISNet3D-synth	**94.36**	91.67	**93.00**	**88.56**	71.90	**91.85**	**80.66**	**68.63**	63.35	**81.87**	**71.43**	**54.47**	62.69	**69.07**	**65.73**	**51.60**

Table 3 Comparison of object-based evaluation results for microscopy dataset $\mathcal{V}_5$. The best performance with respect to each metric are in bold

Methods	Microscopy $\mathcal{V}_5$						
	mP	mR	mF_1	$AP_{.50}$	$AP_{.75}$	mAP	AJI
StarDist3D [15]	73.44	74.24	73.84	88.59	9.78	61.20	68.56
DeepSynth [6]	81.63	76.04	78.72	71.47	50.88	63.74	75.54
Cellpose [3]	96.15	94.47	95.30	94.90	83.82	91.21	81.31
NISNet3D-slim	**96.89**	**96.24**	**96.56**	**95.98**	**88.84**	**93.62**	**83.90**

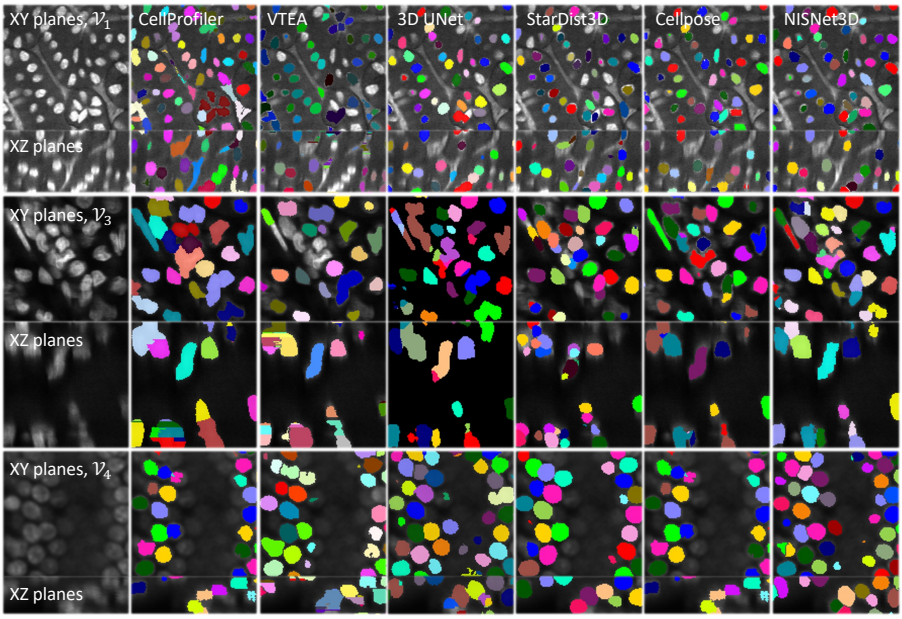

Fig. 5 Visualization of segmentation results for XY and XZ focal planes

For example, $AP_{.25}$ is the average precision with IoU threshold 0.25. In addition, we use the Aggregated Jaccard Index (AJI) [47] to integrate object and voxel errors.

All methods are optimized to achieve the best visual results by parameter tuning. This is further discussed in Section 3.5. The quantitative evaluation results for microscopy datasets $\mathcal{V}_1$-$\mathcal{V}_4$ are shown in Table 2, and the quantitative evaluation results for microscopy dataset $\mathcal{V}_5$ are shown in Table 3 due to different IoU thresholds. We evaluate all subvolumes together for each dataset. In other words, for all subvolumes in a dataset, there is only one mP, mR, F_1, AP, and mAP for each dataset. We noticed 3D Watershed, Squassh, Cellprofiler, and VTEA generate segmentation results with more heterogeneity for different subvolumes of each dataset. Figure 4 shows the AP scores using multiple IoU thresholds and the box plot of AJI on each subvolume of dataset

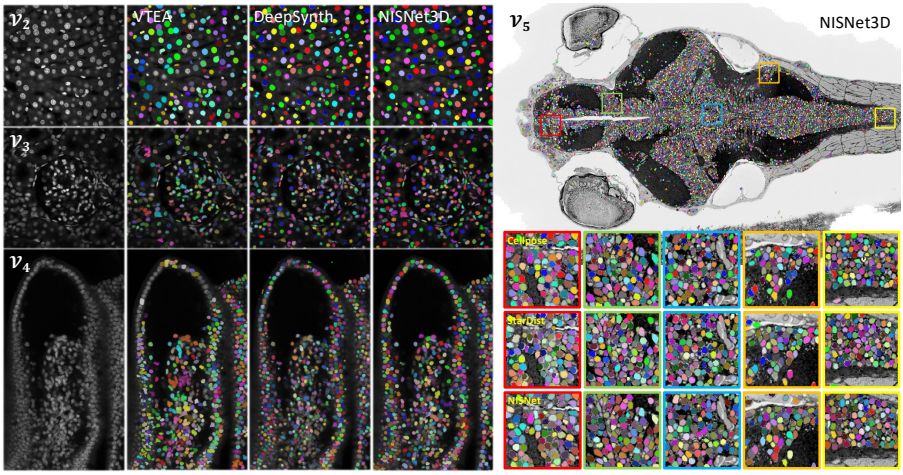

Fig. 6 Visual comparison of the segmentations on a slice of the entire volume for datasets $\mathcal{V}_2$, $\mathcal{V}_3$, $\mathcal{V}_4$, and $\mathcal{V}_5$. Note that datasets $\mathcal{V}_2$, $\mathcal{V}_3$, and $\mathcal{V}_4$ are fluorescence microscopy volumes and dataset $\mathcal{V}_5$ is electron microscopy volume of zebrafish brain

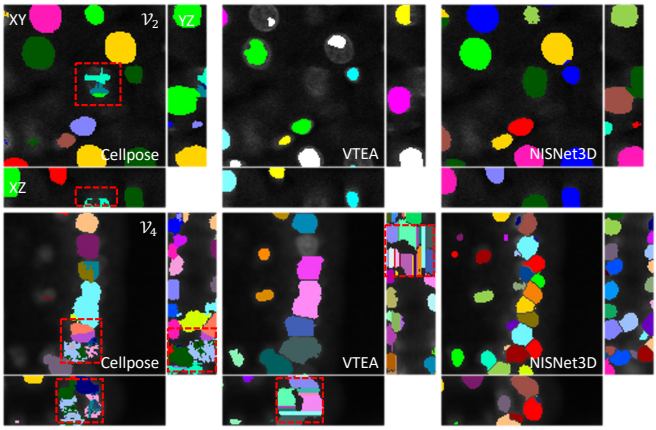

Fig. 7 Visualization of segmentation results of Cellpose and VTEA compared with NISNet3D. The red boxes show the segmentation differences. Cellpose struggles reconstructing 3D segmentation from 2D segmentation results, leading to incomplete segmented 3D objects with holes when viewing from 3D perspective. VTEA struggles to distinguish different objects, resulting in under-segmentation. Also, the VTEA reconstruction results show strong artifacts for different slices of each 3D object.

$\mathcal{V}_2$. The orthogonal views (XY focal planes and XZ focal planes) of the segmentation masks that are overlaid on the original microscopy subvolume for each method on $\mathcal{V}_1$-$\mathcal{V}_5$ are shown in Figure 5 and Figure 6. Note the colors correspond to different nuclei.

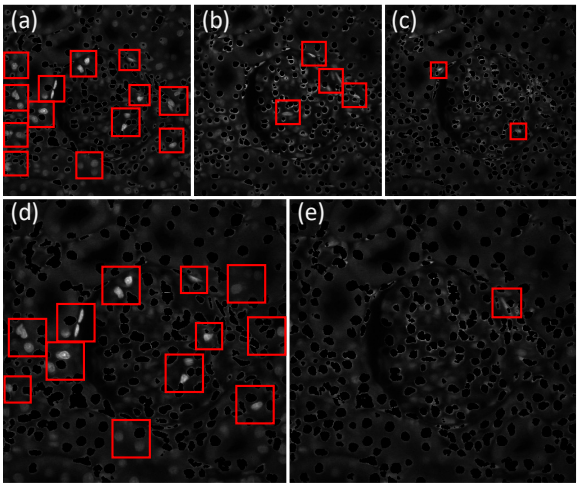

Fig. 8 (a)(b)(c) shows the *Difference Volume* for VTEA, DeepSynth, and NISNet3D, respectively, using *Visualization Method A*. The nuclei voxels shown are the nuclei voxels in the original microscopy volume that are not segmented. (d)(e) shows the *Difference Volume* for VTEA and DeepSynth respectively, using *Visualization Method B*. The nuclei shown are the nuclei segmented by NISNet3D but completely missed by VTEA and DeepSynth, respectively. The red bounding boxes show the nuclei that are not segmented correctly

2.5 Visualizing Differences

Entire microscopy volumes contain many regions with varying spatial characteristics. In order to see how the segmentation methods perform on various regions, Figure 7 shows the 2D to 3D reconstruction error from Cellpose and VTEA compared with NISNet3D. In addition, we use three methods for visualizing the differences between a "test segmented volume" and a "reference segmented volume". The first method *Visualization Method A* shows the false negative voxels of a segmented volume, the second method *Visualization Method B* shows the undersegmentation regions where multiple nuclei in the "reference segmented volume" are detected as a single nucleus in the "test segmented volume", and the second method *Visualization Method C* shows the false-negative objects by comparing with a "reference segmented volume". The details of the three methods are described in Section 3.7. We use the NISNet3D segmented volume as the "reference segmented volume". We visualize the segmentation differences between VTEA and NISNet3D and we also visualize the differences between DeepSynth and NISNet3D on entire volumes. Figure 8 shows the *Difference Volume* for VTEA, DeepSynth, and NISNet3D using *Visualization Method A* and *C*. Figure 9 shows the *Difference Volume* for using *Visualization Method B*.

2.6 Discussion

In this paper, we describe a true 3D Nuclei Instance Segmentation Network, known as NISNet3D, for fluorescence microscopy images segmentation. Instead

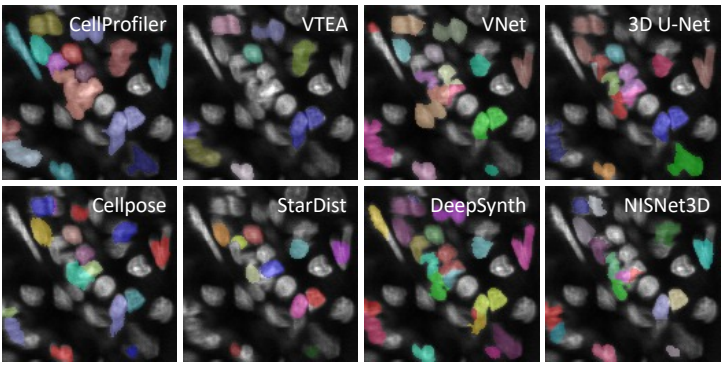

Fig. 9 *Difference Volume* for dataset $\mathcal{V}_3$ using *Visualization Method B* which shows the undersegmentation regions where multiple nuclei in the "reference segmented volume" are detected as a single nucleus in the "test segmented volume". The ground truth annotated volume is used as the "reference segmented volume" here

of segmenting 2D images slice by slice and reconstruct to 3D segmentation results, our approach directly works on 3D volumes by making use of a modified 3D U-Net and a nuclei instance segmentation system for separating touching nuclei. We improved the classical distance transform used for marker-controlled watershed by generating the 3D gradient volume and further obtain more accurate markers for each nuclei. Both visual and quantitative evaluation results demonstrate that directly using 3D CNN can improve the segmentation accuracy.

Due to the limited ground truth data, we use the synthetic microscopy volumes generated from SpCycleGAN for training. We improved the synthetic image generation by modeling nuclei as deformed ellipsoids. NISNet3D can be trained on both actual microscopy volumes and synthetic microscopy volumes generated using SpCycleGAN or a combination of both. NISNet3D can also be trained on synthetic data and further lightly retrained on limited number of other types of microscopy data as an incremental improvement. This can significantly reduce the computational resources needed for biologists since they only need an inference engine and update it with few annotated volumes. We demonstrate that NISNet3D performs well when compared to other methods on a variety of microscopy data both visually and quantitatively. To better inspect the segmentation results, we also present three visualization methods for visualizing segmentation differences in large 3D microscopy volumes without the need of ground truth annotations. Visualizing segmentation differences in different regions of a large microscopy volume is important for the understanding of model performance.

In the future, we will explore more synthetic nuclei segmentation masks generation methods that can better simulate the complex nuclei structures with more realistic density and distributions. We will extend the current SpCycleGAN to a 3D version such that it can directly generate 3D microscopy volumes with point spread function incorporated. In this way, the generated

synthetic volumes will be more close to the distribution of real microscopy volumes, and can further improve the NISNet3D segmentation accuracy.

References

[1] Lucas, A.M., Ryder, P.V., Li, B., Cimini, B.A., Eliceiri, K.W., Carpenter, A.E.: Open-source deep-learning software for bioimage segmentation. Molecular Biology of the Cell **32**(9), 823–829 (2021)

[2] Piccinini, F., Balassa, T., Carbonaro, A., Diosdi, A., Toth, T., Moshkov, N., Tasnadi, E.A., Horvath, P.: Software tools for 3d nuclei segmentation and quantitative analysis in multicellular aggregates. Computational and structural biotechnology journal **18**, 1287–1300 (2020)

[3] Stringer, C., Wang, T., Michaelos, M., Pachitariu, M.: Cellpose: a generalist algorithm for cellular segmentation. Nature Methods **18**(1), 100–106 (2021)

[4] Greenwald, N.F., Miller, G., Moen, E., Kong, A., Kagel, A., Dougherty, T., Fullaway, C.C., McIntosh, B.J., Leow, K.X., Schwartz, M.S., *et al.*: Whole-cell segmentation of tissue images with human-level performance using large-scale data annotation and deep learning. Nature biotechnology **40**(4), 555–565 (2022)

[5] Kromp, F., Fischer, L., Bozsaky, E., Ambros, I.M., Dörr, W., Beiske, K., Ambros, P.F., Hanbury, A., Taschner-Mandl, S.: Evaluation of deep learning architectures for complex immunofluorescence nuclear image segmentation. IEEE Transactions on Medical Imaging **40**(7), 1934–1949 (2021)

[6] Dunn, K.W., Fu, C., Ho, D.J., Lee, S., Han, S., Salama, P., Delp, E.J.: Deepsynth: Three-dimensional nuclear segmentation of biological images using neural networks trained with synthetic data. Scientific Reports **9**(1), 18295–18309 (2019)

[7] Fu, C., Lee, S., Ho, D.J., Han, S., Salama, P., Dunn, K.W., Delp, E.J.: Three dimensional fluorescence microscopy image synthesis and segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2302–2310 (2018). Salt Lake City, UT

[8] Yushkevich, P.A., Piven, J., Hazlett, H.C., Smith, R.G., Ho, S., Gee, J.C., Gerig, G.: User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability. Neuroimage **31**(3), 1116–1128 (2006)

[9] Berger, D.R., Seung, H.S., Lichtman, J.W.: Vast (volume annotation and segmentation tool): efficient manual and semi-automatic labeling of large

3d image stacks. Frontiers in neural circuits **12**, 88 (2018)

[10] Hollandi, R., Diósdi, Á., Hollandi, G., Moshkov, N., Horváth, P.: Annotatorj: an imagej plugin to ease hand annotation of cellular compartments. Molecular biology of the cell **31**(20), 2179–2186 (2020)

[11] Borland, D., McCormick, C.M., Patel, N.K., Krupa, O., Mory, J.T., Beltran, A.A., Farah, T.M., Escobar-Tomlienovich, C.F., Olson, S.S., Kim, M., *et al.*: Segmentor: a tool for manual refinement of 3d microscopy annotations. BMC bioinformatics **22**(1), 1–12 (2021)

[12] Wang, J., Perez, L.: The effectiveness of data augmentation in image classification using deep learning. arXiv preprint arXiv:1712.04621 (2017)

[13] Mikołajczyk, A., Grochowski, M.: Data augmentation for improving deep learning in image classification problem. International Interdisciplinary PhD Workshop, 117–122 (2018). Swinoujscie, Poland

[14] Yang, L., Ghosh, R.P., Franklin, J.M., Chen, S., You, C., Narayan, R.R., Melcher, M.L., Liphardt, J.T.: Nuset: A deep learning tool for reliably separating and analyzing crowded cells. PLoS computational biology **16**(9), 1008193 (2020)

[15] Weigert, M., Schmidt, U., Haase, R., Sugawara, K., Myers, G.: Star-convex polyhedra for 3d object detection and segmentation in microscopy. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 3666–3673 (2020). Snowmass, CO

[16] Sadanandan, S.K., Ranefall, P., Le Guyader, S., Wählby, C.: Automated training of deep convolutional neural networks for cell segmentation. Scientific reports **7**(1), 1–7 (2017)

[17] Caicedo, J.C., Roth, J., Goodman, A., Becker, T., Karhohs, K.W., Broisin, M., Molnar, C., McQuin, C., Singh, S., Theis, F.J., *et al.*: Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. Cytometry Part A **95**(9), 952–965 (2019)

[18] Baniukiewicz, P., Lutton, E.J., Collier, S., Bretschneider, T.: Generative adversarial networks for augmenting training data of microscopic cell images. Frontiers in Computer Science, 10 (2019)

[19] Wiesner, D., Nečasová, T., Svoboda, D.: On generative modeling of cell shape using 3d gans. International Conference on Image Analysis and Processing, 672–682 (2019)

[20] Chen, A., Wu, L., Han, S., Salama, P., Dunn, K.W., Delp, E.J.: Three dimensional synthetic non-ellipsoidal nuclei volume generation using

bezier curves. Proceedings of the IEEE International Symposium on Biomedical Imaging (2021). Nice, France

[21] Wu, L., Han, S., Chen, A., Salama, P., Dunn, K.W., Delp, E.J.: Rcnn-slicenet: A slice and cluster approach for nuclei centroid detection in three-dimensional fluorescence microscopy images. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 3750–3760 (2021). Nashville, TN

[22] Wu, L., Chen, A., Salama, P., Dunn, K.W., Delp, E.J.: An ensemble learning and slice fusion strategy for three-dimensional nuclei instance segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (2022). New Orleans, LA

[23] Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: A survey. International journal of computer vision **128**(2), 261–318 (2020)

[24] Carneiro, G., Zheng, Y., Xing, F., Yang, L.: Review of deep learning methods in mammography, cardiovascular, and microscopy image analysis. Deep Learning and Convolutional Neural Networks for Medical Image Computing, 11–32 (2017)

[25] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. Medical Image Computing and Computer-Assisted Intervention **9351**, 231–241 (2015). Munich, Germany

[26] Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3D u-net: Learning dense volumetric segmentation from sparse annotation. Medical Image Computing and Computer-Assisted Intervention **9901**, 424–432 (2016)

[27] Milletari, F., Navab, N., Ahmadi, S.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. International Conference on 3D Vision, 565–571 (2016)

[28] Balakrishnan, G., Zhao, A., Sabuncu, M.R., Guttag, J., Dalca, A.V.: Voxelmorph: A learning framework for deformable medical image registration. IEEE Transactions on Medical Imaging **38**(8), 1788–1800 (2019)

[29] Graham, S., Vu, Q.D., Raza, S.E.A., Azam, A., Tsang, Y.W., Kwak, J.T., Rajpoot, N.: Hover-net: Simultaneous segmentation and classification of nuclei in multi-tissue histology images. Medical Image Analysis **58**, 101563 (2019)

[30] Fu, C., Ho, D.J., Han, S., Salama, P., Dunn, K.W., Delp, E.J.: Nuclei

segmentation of fluorescence microscopy images using convolutional neural networks. Proceedings of the IEEE International Symposium on Biomedical Imaging, 704–708 (2017). Melbourne, Australia

[31] Ho, D.J., Fu, C., Salama, P., Dunn, K.W., Delp, E.J.: Nuclei segmentation of fluorescence microscopy images using three dimensional convolutional neural networks. IEEE Conference on Computer Vision and Pattern Recognition Workshops, 834–842 (2017)

[32] Schmidt, U., Weigert, M., Broaddus, C., Myers, G.: Cell detection with star-convex polygons. Medical Image Computing and Computer Assisted Intervention, 265–273 (2018). Granada, Spain

[33] Ho, D.J., Montserrat, D.M., Fu, C., Salama, P., Dunn, K.W., Delp, E.J.: Sphere estimation network: three-dimensional nuclei detection of fluorescence microscopy images. Journal of Medical Imaging **7**(4), 1–16 (2020)

[34] Mandal, S., Uhlmann, V.: Splinedist: Automated cell segmentation with spline curves. Proceedings of the International Symposium on Biomedical Imaging, 1082–1086 (2021). Nice, France

[35] Simard, P.Y., Steinkraus, D., Platt, J.C.: Best practices for convolutional neural networks applied to visual document analysis. Proceedings of the International Conference on Document Analysis and Recognition, 958–963 (2003). Edinburgh, UK

[36] Zhu, J., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. Proceedings of the IEEE International Conference on Computer Vision, 2242–2251 (2017). Venice, Italy

[37] Chen, A., Wu, L., Winfree, S., Dunn, K.W., Salama, P., Delp, E.J.: 3d ground truth annotations of nuclei in 3d microscopy volumes. bioRxiv (2022)

[38] Lin, Z., Wei, D., Petkova, M.D., Wu, Y., Ahmed, Z., Zou, S., Wendt, N., Boulanger-Weill, J., Wang, X., Dhanyasi, N., et al.: Nucmm dataset: 3d neuronal nuclei instance segmentation at sub-cubic millimeter scale. International Conference on Medical Image Computing and Computer-Assisted Intervention, 164–174 (2021)

[39] Yang, X., Li, H., Zhou, X.: Nuclei segmentation using marker-controlled watershed, tracking using mean-shift, and kalman filter in time-lapse microscopy. IEEE Transactions on Circuits and Systems I: Regular Papers **53**(11), 2405–2414 (2006)

[40] Rizk, A., Paul, G., Incardona, P., Bugarski, M., Mansouri, M., Niemann, A., Ziegler, U., Berger, P., Sbalzarini, I.F.: Segmentation and quantification of subcellular structures in fluorescence microscopy images using squassh. Nature Protocols **9**(3), 586–596 (2014)

[41] McQuin, C., Goodman, A., Chernyshev, V., Kamentsky, L., Cimini, B.A., Karhohs, K.W., Doan, M., Ding, L., Rafelski, S.M., Thirstrup, D., Wiegraebe, W., Singh, S., Becker, T., Caicedo, J.C., Carpenter, A.E.: Cellprofiler 3.0: Next-generation image processing for biology. PLoS biology **16**(7), 2005970–117 (2018)

[42] Winfree, S., Khan, S., Micanovic, R., Eadon, M.T., Kelly, K.J., Sutton, T.A., Phillips, C.L., Dunn, K.W., El-Achkar, T.M.: Quantitative three-dimensional tissue cytometry to study kidney tissue and resident immune cells. Journal of the American Society of Nephrology **28**(7), 2108–2118 (2017)

[43] Hosang, J., Benenson, R., Dollár, P., Schiele, B.: What makes for effective detection proposals? IEEE Transactions on Pattern Analysis and Machine Intelligence **38**(4), 814–830 (2016)

[44] Everingham, M., Eslami, S.M.A., Gool, L.V., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision **111**(1), 98–136 (2015)

[45] Padilla, R., Netto, S.L., da Silva, E.A.B.: A survey on performance metrics for object-detection algorithms. Proceedings of the International Conference on Systems, Signals and Image Processing, 237–242 (2020). Niteroi, Brazil

[46] Davis, J., Goadrich, M.: The relationship between precision-recall and roc curves. Proceedings of the international conference on Machine learning, 233–240 (2006). Pittsburgh, PA

[47] Kumar, N., Verma, R., Sharma, S., Bhargava, S., Vahadane, A., Sethi, A.: A dataset and a technique for generalized nuclear segmentation for computational pathology. IEEE Transactions on Medical Imaging **36**(7), 1550–1560 (2017)

[48] McKinley, S., Levine, M.: Cubic spline interpolation. College of the Redwoods **45**(1), 1049–1060 (1998)

[49] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. IEEE Conference on Computer Vision and Pattern Recognition, 770–778 (2016). Venice, Italy

[50] Oktay, O., Schlemper, J., Folgoc, L.L., Lee, M., Heinrich, M., Misawa,

K., Mori, K., McDonagh, S., Hammerla, N.Y., Kainz, B., Glocker, B., Rueckert, D.: Attention u-net: Learning where to look for the pancreas. arXiv preprint arXiv:1804.03999 (2018)

[51] Han, S., Lee, S., Fu, C., Salama, P., Dunn, K., Delp, E.J.: Nuclei counting in microscopy images with three dimensional generative adversarial networks. Proceedings of the SPIE Conference on Medical Imaging **10949**, 753–763 (2019). International Society for Optics and Photonics. San Diego, CA

[52] Zhang, D., Song, Y., Liu, S., Feng, D., Wang, Y., Cai, W.: Nuclei instance segmentation with dual contour-enhanced adversarial network. Proceedings of the IEEE International Symposium on Biomedical Imaging, 409–412 (2018). Washington, DC

[53] Lin, T., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. Proceedings of the IEEE International Conference on Computer Vision, 2999–3007 (2017). Venice, Italy

[54] Salehi, M., Sadegh, S., Erdogmus, D., Gholipour, A.: Tversky loss function for image segmentation using 3d fully convolutional deep networks. Proceedings of International Workshop on Machine Learning in Medical Imaging, 379–387 (2017). Quebec, Canada

[55] Soille, P., Vincent, L.M.: Determining watersheds in digital pictures via flooding simulations. Visual Communications and Image Processing'90: Fifth in a Series **1360**, 240–250 (1990). International Society for Optics and Photonics

[56] Di Stefano, L., Bulgarelli, A.: A simple and efficient connected components labeling algorithm. Proceedings of the International Conference on Image Analysis and Processing, 322–327 (1999). Venice, Italy

[57] Santella, A., Du, Z., Nowotschin, S., Hadjantonakis, A.K., Bao, Z.: A hybrid blob-slice model for accurate and efficient detection of fluorescence labeled nuclei in 3d. BMC bioinformatics **11**(1), 1–13 (2010)

[58] Isola, P., Zhu, J., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. Proceedings of the IEEE conference on computer vision and pattern recognition, 5967–5976 (2017). Honolulu, HI

[59] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2017)

[60] Lin, T., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. IEEE International Conference on Computer Vision, 2999–3007 (2017). Venice, Italy

[61] Rahman, M.A., Wang, Y.: Optimizing intersection-over-union in deep neural networks for image segmentation. International symposium on visual computing, 234–244 (2016). Lake Tahoe, NV

[62] Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S.: Generalized intersection over union: A metric and a loss for bounding box regression. Proceedings of the IEEE conference on computer vision and pattern recognition, 658–666 (2019). Long Beach, CA

[63] Powers, D.M.: Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061 (2020)

3 METHOD

The block diagram of proposed nuclei instance segmentation system is shown in Figure 10, which includes: (1) 3D microscopy image synthesis and annotated data, (2) NISNet3D training and inference, and (3) 3D nuclei instance segmentation.

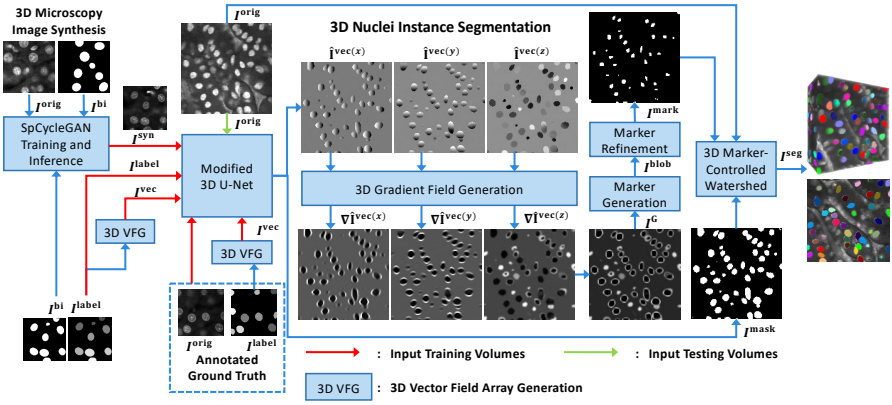

Fig. 10 Overview of NISNet3D for 3D nuclei instance segmentation. Note: The training and synthetic image generation are also shown but are not explicitly part of NISNet3D. NISNet3D is trained with synthetic microscopy volumes generated from SpCycleGAN and/or with annotated actual volumes as indicated

3.1 Notation and Overview

In this paper, we denote a 3D image volume of size $X \times Y \times Z$ voxels by I, and a voxel having coordinates (x, y, z) in the volume by $I_{(x,y,z)}$. We will use superscripts to distinguish between the different types of volumes and arrays needed to achieve this goal. For example, I^{orig}, I^{bi}, and I^{syn} will be used to denote an original microscopy volume, a volume of binary segmentation masks, and a synthetic microscopy volume, respectively. The objective is to segment nuclei within I^{orig} using synthetic data. In addition, the various nuclei corresponding to the different masks in I^{bi} are marked with unique voxel intensities and comprise the volume I^{label}. This is done to distinguish each nuclei instance and will be used during the training phase.

In addition to I^{label}, NISNet3D generates from I^{label} a "vector field array," I^{vec} of size $X \times Y \times Z \times 3$, also for training purposes. Each element $I^{\text{vec}}_{(x,y,z)}$ of I^{vec}, located at (x, y, z), is a 3D vector $\vec{\mathbf{V}}_{(x,y,z)}$ that is associated with the nucleus voxel $I^{\text{label}}_{(x,y,z)}$ and points to the centroid of the nucleus with which it is associated. Detailed description of vector field generation will be discussed in Section 3.3.1.

For a given input volume, I^{orig}, NISNet3D generates a corresponding volume I^{mask} of size $X \times Y \times Z \times 3$ comprising the binary segmentation results as

well as an estimate of the vector field array, $\hat{I}^{\text{vec}}$, which is used to locate the nuclei centroids. In addition, $\hat{I}^{\text{vec}}$ is used to generate a 3D array of gradients, I^{G} of size $X \times Y \times Z$, that is used in conjunction with morphological operations and watershed segmentation to distinguish and separate touching nuclei as well as remove small objects. The final output, I^{seg}, is a color-coded segmentation volume. Figure 10 provides an overview of our proposed approach.

3.2 3D Nuclei Image Synthesis

Deep learning methods generally require large amounts of training samples to achieve accurate results. However, manually annotating ground truth is a tedious task and impractical in many situations especially for 3D microscopy volumes. To address this, NISNet3D uses synthetic 3D microscopy volumes for training a segmentation network. It must be emphasized that NISNet3D can use synthetic volumes, annotated real volumes, or combinations of both. We demonstrate this in the experiments.

To generate synthetic volumes, we first generate synthetic segmentation masks that are used as ground truth masks, and then translate the synthetic segmentation masks into synthetic microscopy volumes using an unsupervised image-to-image translation model known as SpCycleGAN [7].

3.2.1 Synthetic Segmentation Mask Generation

To generate a volume of synthetic binary nuclei segmentation masks, we iteratively add N initial binary nuclei to an empty 3D volume of size $128 \times 128 \times 128$. Each initial nuclei is modeled as a 3D binary ellipsoid having random size, orientation, and location. The size, orientation, and location of the nth ellipsoid are parameterized by $\boldsymbol{a}^{(n)}$, $\boldsymbol{\theta}^{(n)}$, and $\boldsymbol{t}^{(n)}$, respectively, where $\boldsymbol{a}^{(n)} = [a_x^{(n)}, a_y^{(n)}, a_z^{(n)}]^T$ denotes a vector of semi-axes lengths of the nth ellipsoid, $\boldsymbol{\theta}^{(n)} = [\theta_x^{(n)}, \theta_y^{(n)}, \theta_z^{(n)}]^T$ denotes a vector of rotation angles relative to the x, y, and z coordinate axes, and $\boldsymbol{t}^{(n)} = [t_x^{(n)}, t_y^{(n)}, t_z^{(n)}]^T$ denotes the displacement/location vector of the nth ellipsoid relative the origin. The parameters $\boldsymbol{a}^{(n)}$, $\boldsymbol{\theta}^{(n)}$, and $\boldsymbol{t}^{(n)}$ are randomly selected based on observations of nuclei characteristics in actual microscopy volumes.

Let $J^{\text{nuc},(n)}_{(x,y,z)}$ denote the nth initial nucleus at location (x, y, z) having intensity/label $n \in \{1, ..., N\}$, then

$$J^{\text{nuc},(n)}_{(x,y,z)} = \begin{cases} n, & \text{if } (\frac{x}{a_x^{(n)}})^2 + (\frac{y}{a_y^{(n)}})^2 + (\frac{z}{a_z^{(n)}})^2 < 1 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The ellipsoids are assigned different intensity values to differentiate them from each other. Subsequently, each $J^{\text{nuc},(n)}_{(x,y,z)}$ undergoes a random translation given by $\boldsymbol{t}^{(n)}$ followed by random rotations specified in $\boldsymbol{\theta}^{(n)}$. Denoting the original coordinates by $\mathbf{X}$ and the translated and rotated coordinates by $\tilde{\mathbf{X}}$, respectively

then

$$\tilde{\mathbf{X}} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{bmatrix} = R_z(\theta_z^{(n)}) R_y(\theta_y^{(n)}) R_x(\theta_x^{(n)}) \begin{bmatrix} x + t_x^{(n)} \\ y + t_y^{(n)} \\ z + t_z^{(n)} \end{bmatrix} \tag{2}$$

where $R_x(\theta_x^{(n)})$, $R_y(\theta_y^{(n)})$ and $R_z(\theta_z^{(n)})$ denote the rotation matrices relative to the x, y, z axes by angles $\theta_x^{(n)}$, $\theta_y^{(n)}$, and $\theta_z^{(n)}$, respectively. Thus, the final n^{th} ellipsoid $I_{(x,y,z)}^{\text{nuc},(n)}$ is given by $I_{(x,y,z)}^{\text{nuc},(n)} = J_{(\tilde{x},\tilde{y},\tilde{z})}^{\text{nuc},(n)}$ for all $n \in \{1, ..., N\}$. We finally constrain the transformed nuclei such that they do not overlap by more than t_{ov} voxels. The ellipsoids $I_{(x,y,z)}^{\text{nuc},(n)}$, $n \in \{1, ..., N\}$, are then used to fill in the binary volume I^{bi} of size $X \times Y \times Z$.

Since actual nuclei are not strictly ellipsoidal but look more like deformed ellipsoids (See Figure 2 (left column)), we use an elastic transformation [35] to deform I^{bi}. To achieve this we first generate a "coarse displacement vector field", I^{coarse}, which is an array of size $d \times d \times d \times 3$, whose entries are independent and Gaussian distributed random variables having zero mean and variance σ^2 (that is $I_{(k,x,y,z)}^{\text{coarse}} \sim \mathcal{N}(0, \sigma^2)$). The size parameter d is used to control the amount of deformation applied to the nuclei. I^{coarse} is then interpolated to size $X \times Y \times Z \times 3$, via spline interpolation [48] or bilinear interpolation [35] to produce a "smooth displacement vector field" I^{smooth}. The entry $I_{(x,y,z,1)}^{\text{smooth}}$ indicates the distance by which voxel $I_{(x,y,z)}^{\text{bi}}$ will be shifted along the x-axis. Similarly, the elements $I_{(x,y,z,2)}^{\text{smooth}}$ and $I_{(x,y,z,3)}^{\text{smooth}}$ provide the distances by which $I_{(x,y,z)}^{\text{bi}}$ will be shifted along the y and z-axes, respectively. In our experiments, we used spline interpolation and values for $d \in \{4, 5, 10\}$. Each element in I^{coarse} serves as an anchor point controlling the magnitude of elastic transform. Larger values of d result in larger deformations being applied through I^{smooth} whereas smaller d values produce less deformation. Examples of deformed ellipsoids are shown in Figure 2 (right column).

3.2.2 Synthetic Microscopy Volume Generation

In the previous section we described how we generate binary segmentation masks. In the current section we describe how we use the masks to generate synthetic microscopy volumes. In particular, we use the unpaired image-to-image translation model known as SpCycleGAN [7, 36] for generating synthetic microscopy volumes. By unpaired we mean that the inputs to SpCycleGAN are volumes of binary segmentation masks and actual microscopy images that do not correspond with each other. In addition, the binary segmentation masks created above are not used as ground truth volumes for performing segmentation of actual microscopy images.

As shown in Figure 10, we use slices from volumes of the binary segmentation masks, $I^{\text{bi},(m)}, m \in \{1, ..., M\}$, where M is the number of training samples/volumes, and slices from actual microscopy volumes $I^{\text{orig},(m)}, m \in$

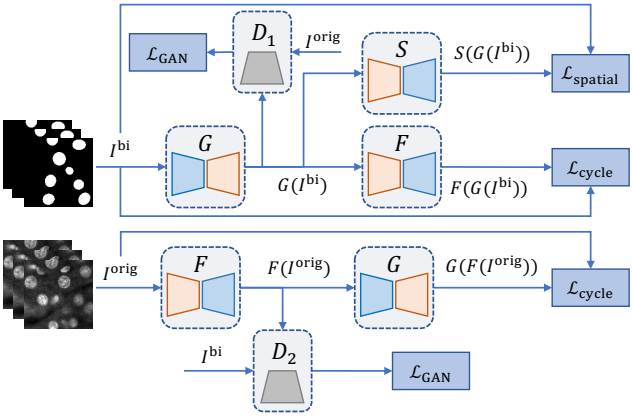

Fig. 11 The architecture of SpCycleGAN that is used for generating synthetic microscopy volumes

$\{1, ..., M\}$ for training SpCycleGAN. After training we generate L synthetic microscopy volumes, $I^{\text{syn},(l)}, l \in \{1, ..., L\}$, using synthetic microscopy segmentation masks, $I^{\text{bi},(l)}, l \in \{1, ..., L\}$ other than the ones used for training. Note that since SpCycleGAN generates 2D slices, we use the slices to construct a 3D synthetic volume.

SpCycleGAN [7], an extension of CycleGAN [36], is shown in Figure 11. SpCycleGAN consists of two generators G and F, two discriminators D_1 and D_2. G learns the mapping from I^{orig} to I^{bi} whereas F performs the reverse mapping. Also, SpCycleGAN introduced a segmentor S for maintaining the spatial location between I^{bi} and $F(G(I^{\text{bi}}))$. The entire loss function of SpCycleGAN is shown in Equation 3.

$$\begin{aligned}\mathcal{L}(G, F, S, D_1, D_2) &= \mathcal{L}_{\text{GAN}}(G, D_1, I^{\text{bi}}, I^{\text{orig}}) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_2, I^{\text{orig}}, I^{\text{bi}}) \\ &+ \lambda_1 \mathcal{L}_{\text{cycle}}(G, F, I^{\text{orig}}, I^{\text{bi}}) \\ &+ \lambda_2 \mathcal{L}_{\text{spatial}}(G, S, I^{\text{orig}}, I^{\text{bi}})\end{aligned} \tag{3}$$

where λ_1 and λ_2 are weight coefficients controlling the loss balance between $\mathcal{L}_{\text{cycle}}$ and $\mathcal{L}_{\text{spatial}}$, and

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, D_1, I^{\text{bi}}, I^{\text{orig}}) &= \mathbb{E}_{I^{\text{orig}}}[\log(D_1(I^{\text{orig}}))] \\ &+ \mathbb{E}_{I^{\text{bi}}}[\log(1 - D_1(G(I^{\text{bi}})))] \\ \mathcal{L}_{\text{GAN}}(F, D_2, I^{\text{orig}}, I^{\text{bi}}) &= \mathbb{E}_{I^{\text{bi}}}[\log(D_2(I^{\text{bi}}))] \\ &+ \mathbb{E}_{I^{\text{orig}}}[\log(1 - D_2(F(I^{\text{orig}})))] \\ \mathcal{L}_{\text{cycle}}(G, F, I^{\text{orig}}, I^{\text{bi}}) &= \mathbb{E}_{I^{\text{bi}}}[\|F(G(I^{\text{bi}})) - I^{\text{bi}}\|_1] \\ &+ \mathbb{E}_{I^{\text{orig}}}[\|G(F(I^{\text{orig}})) - I^{\text{orig}}\|_1] \\ \mathcal{L}_{\text{spatial}}(G, S, I^{\text{bi}}, I^{\text{orig}}) &= \mathbb{E}_{I^{\text{bi}}}[\|S(G(I^{\text{bi}})) - I^{\text{bi}}\|_2]\end{aligned} \tag{4}$$

where $\|\cdot\|_1$ and $\|\cdot\|_2$ denotes the L_1 norm and L_2 norm, respectively. $\mathbb{E}_I$ is the expected value over all the input volumes of a batch to the network.

3.3 NISNet3D

NISNet3D is based on a modified 3D U-Net architecture. In this section, we describe the architecture, how to train and inference using the modified 3D U-Net, as well as nuclei instance segmentation performed by NISNet3D. Figure 12 provides an overview of NISNet3D.

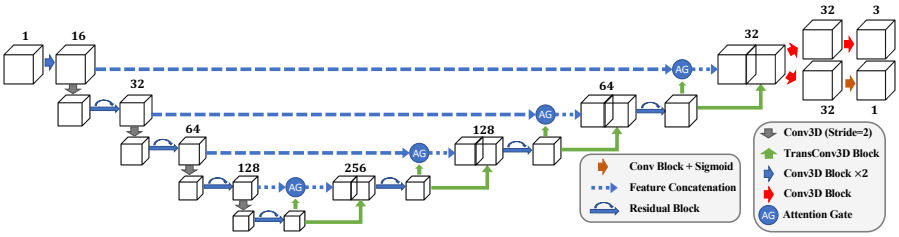

Fig. 12 The proposed NISNet3D uses a modified 3D U-Net architecture with residual blocks, attention gates, and multi-task learning module

3.3.1 Modified 3D U-Net

As mentioned previously NISNet3D utilizes a modified 3D U-Net. In this section we describe its architecture given in Figure 12. In general NISNet3D employs an encoder-decoder network (See Figure 12) which outputs the same size array as the input. The encoder consists of multiple Conv3D Blocks (Figure 13(a)) and Residual Blocks (Figure 13(b)) [49]. Instead of using max pooling layers, we use Conv3D Blocks with stride 2 for feature down-sampling, which introduces more learnable parameters. Each convolution block consists of a 3D convolution layer with filter size $3 \times 3 \times 3$, a 3D batch normalization layer, and a leaky ReLU layer. The decoder consists of multiple TransConv3D blocks (Figure 13(d)) and attention gates (Figure 13(c)). Each TransConv3D block includes a 3D transpose convolution with filter size $3 \times 3 \times 3$ followed by 3D batch normalization and leaky ReLU. We use a self-attention mechanism described in [50] to refine the feature concatenation while reconstructing the spatial information.

Training The Modified 3D U-Net. The modified 3D U-Net can be trained on both synthetic microscopy volumes I^{syn} or actual microscopy volumes I^{orig} if manual ground truth annotations are available.

As shown in Figure 10, during training, NISNet3D's modified 3D U-Net takes an I^{syn} or I^{orig}, an I^{label}, and an I^{vec} as input. I^{label} is the $X \times Y \times Z$ gray-scale label volume corresponding to I^{bi} where different nuclei are marked with unique pixel intensities. It used is used by NISNet3D to learn the segmentation masks. Moreover, to facilitate segmentation NISNet3D identifies nuclei centroids and estimates their locations using a "vector field," I^{vec}. In particular,

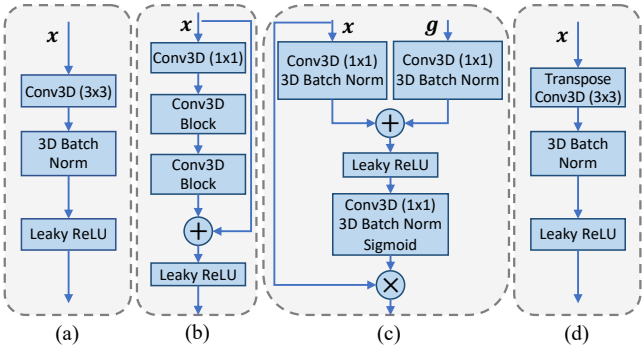

Fig. 13 (a) Conv3D Block, (b) Residual Block, (c) Attention Gate, (d) TransConv3D Block

I^{vec} is part of the ground truth data used during training to enable the network to learn nuclei centroid locations. In addition, the 3D vector field array helps identify the boundaries of adjacent/touching nuclei since their corresponding vectors tend to point in different directions.

As indicated previously a "vector field," I^{vec}, is an $X \times Y \times Z \times 3$ array whose elements are 3D vectors that point to the centroids of the corresponding nuclei. Specifically, $I^{\text{vec}}_{(x,y,z)}$ is a 3D vector at location (x, y, z) and points to the nearest nucleus centroid. We also denote the $X \times Y \times Z$ array consisting of the first component of each 3D vector by $I^{\text{vec}(x)}$. Similarly, we denote the arrays of the second and third components by $I^{\text{vec}(y)}$ and $I^{\text{vec}(z)}$, respectively, that is $I^{\text{vec}}_{(x,y,z)} = (I^{\text{vec}(x)}_{(x,y,z)}, I^{\text{vec}(y)}_{(x,y,z)}, I^{\text{vec}(z)}_{(x,y,z)})$.

The first step in the 3D vector field array generation (VFG) (see in Figure 14) is to obtain the centroid of each nucleus in I^{label}. We denote the kth nucleus as the set of voxels with intensity k in I^{label}, and the location of the centroid of the kth nucleus by (x_k, y_k, z_k). We define $\vec{\mathbf{V}}^{k}_{(x,y,z)}$ to be the 3D vector from (x, y, z) to (x_k, y_k, z_k) and set $I^{\text{vec}}_{(x,y,z)} = \vec{\mathbf{V}}^{k}_{(x,y,z)}$ as long as the voxel at (x, y, z) is not a background voxel. Note that if a voxel $I^{\text{label}}_{(x,y,z)}$ is a background voxel (i.e. $I^{\text{label}}_{(x,y,z)}$=0), the corresponding entry $I^{\text{vec}}_{(x,y,z)}$ in the vector field array is set to $\mathbf{0}$, as shown in Equation 5.

Using I^{syn} or I^{orig}, I^{label}, and I^{vec}, NISNet3D then outputs an estimate of the 3D vector field array $\hat{I}^{\text{vec}}$ ($\hat{I}^{\text{vec}(x)}$, $\hat{I}^{\text{vec}(y)}$, $\hat{I}^{\text{vec}(z)}$) and the 3D binary segmentation mask volume I^{mask}.

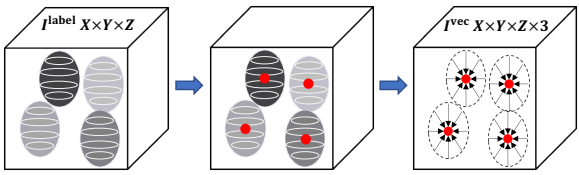

Fig. 14 Steps for 3D vector field array generation (VFG). Each nucleus voxel in the 3D vector field array, I^{vec} represents a 3D vector that points to the centroid of current nucleus

$$I^{\text{vec}}_{(x,y,z)} = \begin{cases} \vec{\mathbf{V}}^{k}_{(x,y,z)}, & \text{if } I^{\text{label}}_{(x,y,z)} \neq 0 \\ \mathbf{0}, & \text{otherwise} \end{cases} \tag{5}$$

Loss Functions. Given I^{syn} or I^{orig}, I^{label}, and I^{vec}, the modified 3D U-Net simultaneously learns the nuclei segmentation masks I^{mask} and the 3D vector field array $\hat{I}^{\text{vec}}$. In the modified 3D U-Net two branches but no sigmoid function are used to obtain $\hat{I}^{\text{vec}}$ because the vector represented at a voxel can point to anywhere in an array. In other words, the entries in $\hat{I}^{\text{vec}}$ can be negative numbers or large numbers. Unlike previous methods [31, 51, 52] that directly learn the distance transform map, the 3D vector field array contains both the distance and direction of the nearest nuclei centroid from the current voxel location. This can help NISNet3D avoid the multiple detection of irregular shaped nuclei. The output 3D vector field array $\hat{I}^{\text{vec}}$ is compared with the ground truth vector field array I^{vec} and the error between them minimized using the Mean Square Error (MSE) loss function. Similarly, the segmentation result I^{mask} is compared with the ground truth binary volume I^{bi} and the difference minimized using a combination of the Focal Loss [53] $\mathcal{L}_{\text{FL}}$ and Tversky Loss [54] $\mathcal{L}_{\text{TL}}$ metrics. Denoting a ground truth binary volume I^{bi} by S, the corresponding segmentation result I^{mask} by $\hat{S}$, a ground truth vector field array I^{vec} by V , and the estimated vector field array $\hat{I}^{\text{vec}}$ by $\hat{V}$, then, the entire loss function is,

$$\begin{aligned} \mathcal{L}(S, \hat{S}, V, \hat{V}) = \lambda_3 \mathcal{L}_{\text{TL}}(S, \hat{S}) + \lambda_4 \mathcal{L}_{\text{FL}}(S, \hat{S}) \\ + \lambda_5 \mathcal{L}_{\text{MSE}}(V, \hat{V}) \end{aligned} \tag{6}$$

where

$$\begin{aligned} \mathcal{L}_{\text{TL}} &= \frac{\sum_{i=1}^{P} s_{i1}\hat{s}_{i1}}{\sum_{i=1}^{P} s_{i1}\hat{s}_{i1} + \alpha_1 \sum_{i=1}^{P} s_{i1}\hat{s}_{i0} + \alpha_2 \sum_{i=1}^{P} s_{i0}\hat{s}_{i1}}, \\ \mathcal{L}_{\text{FL}} &= -\frac{1}{P}\sum_{i=1}^{P} \{\beta s_{i1}\hat{s}_{i0}^{\gamma}\log(\hat{s}_{i1}) + (1-\beta) s_{i0}\hat{s}_{i1}^{\gamma}\log(\hat{s}_{i0})\}, \\ \mathcal{L}_{\text{MSE}} &= \frac{1}{P}\sum_{i=1}^{P} (v_i - \hat{v}_i)^2 \end{aligned} \tag{7}$$

where $\alpha_1 + \alpha_2 = 1$ are two hyper-parameters in Tversky loss [54] that control the balance between false positive and false negative detections. β and γ are two hyper-parameters in Focal loss [53] where β balances the importance of positive/negative voxels, and γ adjusts the weights for easily classified voxels. $v_i \in V$ is the ith element of V, and $\hat{v}_i \in \hat{V}$ is the ith entry in $\hat{V}$. Similarly, $s_i \in S$ is the ith voxel in S, and $\hat{s}_i \in \hat{S}$ is the ith voxel in $\hat{S}$. We define $\hat{s}_{i0}$ to be the probability that the ith voxel in $\hat{S}$ is a nuclei, and $\hat{s}_{i1}$ as the probability

that the ith voxel in $\hat{S}$ is a background voxel. Similarly, $s_{i1} = 1$ if s_i is a nuclei voxel and 0 if s_i is a background voxel, and vice versa for s_{i0}. Lastly, P is the total number of voxels in a volume.

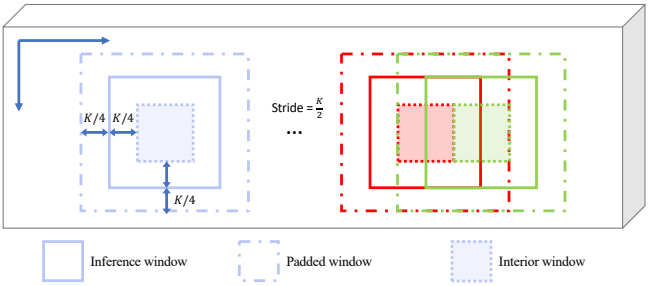

Fig. 15 Proposed divide-and-conquer inference scheme for segmenting large microscopy volumes

Modified 3D U-Net Inference. To segment a large microscopy volume, we propose an divide-and-conquer inference scheme shown in Figure 15. We use an inference window of size $K \times K \times K$ that slides along each original microscopy volume I^{orig} of size $X \times Y \times Z$ and crops a subvolume. Considering that cropping may result in some nuclei being partially included, and that these partially included nuclei which lie on the border of the inference window may cause inaccurate segmentation results, we construct a padded window by symmetrically padding each cropped subvolume by $\frac{K}{4}$ voxels on each border. Also, the stride of the moving window is set to $\frac{K}{2}$ so with every step it slides, it will have $\frac{K}{2}$ voxels overlapping with the previous window. For the inference results of every $K \times K \times K$ window, only the interior and centered $\frac{K}{2} \times \frac{K}{2} \times \frac{K}{2}$ subvolume, which we denoted as the interior window, will be used as the segmentation results. In this paper, we use $K = 128$ for all testing data. Once the inference window slides along the entire volume, a segmentation volume I^{mask} and 3D vector field array $\hat{I}^{\text{vec}}$ of size $X \times Y \times Z$ will be generated. In this way, we can inference on any size input volume, especially very large volumes.

3.3.2 3D Nuclei Instance Segmentation

Based on the output of the modified 3D U-Net, a 3D gradient field and an array of markers are generated and used to separate densely clustered nuclei. Moreover, the markers are refined to attain better separation. As mentioned previously, each element/vector in the estimated 3D vector field array $\hat{I}^{\text{vec}}$ point to the centroid of the nearest nucleus. Most often, the vectors associated with voxels on the boundary of two touching nuclei point in different directions that correspond to the locations of the centroids of the touching nuclei, and hence have a large difference/gradient. We employ such larger gradients to detect the boundaries of touching and also overlapping nuclei.

Let $\nabla \hat{I}^{\text{vec}}$ denote the gradient of $\hat{I}^{\text{vec}}$ which is obtained as given in Equation 8:

$$\begin{aligned}\nabla \hat{I}^{\text{vec}} &= \left[\nabla \hat{I}^{\text{vec}(x)} \;\; \nabla \hat{I}^{\text{vec}(y)} \;\; \nabla \hat{I}^{\text{vec}(z)}\right]^T \\ &= \left[\frac{\partial \hat{I}^{\text{vec}(x)}}{\partial x} \;\; \frac{\partial \hat{I}^{\text{vec}(y)}}{\partial y} \;\; \frac{\partial \hat{I}^{\text{vec}(z)}}{\partial z}\right]^T \\ &= \left[S_x * \hat{I}^{\text{vec}(x)} \;\; S_y * \hat{I}^{\text{vec}(y)} \;\; S_z * \hat{I}^{\text{vec}(z)}\right]^T, \end{aligned} \tag{8}$$

where $\hat{I}^{\text{vec}(x)}$, $\hat{I}^{\text{vec}(y)}$, and $\hat{I}^{\text{vec}(z)}$ are the x, y, and z sub-arrays of the estimated 3D vector field array $\hat{I}^{\text{vec}}$ respectively, S_x, S_y, S_z are 3D Sobel filters, and $*$ is the convolution operator. We also define the gradient map I^{G} as the maximum of the gradient components along the x, y and z directions (see Equation 9).

$$I^{\text{G}} = \max(\nabla \hat{I}^{\text{vec}(x)}, \nabla \hat{I}^{\text{vec}(y)}, \nabla \hat{I}^{\text{vec}(z)}) \tag{9}$$

Elements of I^{G} that correspond to boundaries of touching nuclei have larger values (larger gradients) which can be used to identify individual nuclei. We subsequently threshold I^{G} via a thresholding function $\tau(x, T_{\text{m}})$ where $\tau(x, T_{\text{m}}) = 1$ if $x \geq T_{\text{m}}$ otherwise 0. This step is used to highlight and distinguish boundary voxles of individual nuclei from spurious edges that may arise from finding the gradients. The value of T_{m} can affect the number of voxels delineated as boundary voxels. Larger values of T_{m} result in thinner boundaries whereas smaller values of T_{m} result in a thicker nuclei boundaries. We then subtract the result from the binary segmentation mask I^{mask} obtained from the modified 3D U-Net, and then apply that to a non-linear half wave rectifier/RELU activation function $\sigma(\cdot)$ that sets all negative values to 0. We denote the result as I^{blob} as given in Equation 10. The difference $I^{\text{mask}} - \tau(I^{\text{G}}, T_{\text{m}})$ removes boundary voxles of touching nuclei while emphasizing the interior. Since the difference can be negative in some cases, we use $\sigma(\cdot)$ to eliminate negative values. Hence, I^{blob} highlights the interior regions of nuclei which maybe used as markers to perform watershed segmentation [39, 55].

$$I^{\text{blob}} = \sigma(I^{\text{mask}} - \tau(I^{\text{G}}, T_{\text{m}})) \tag{10}$$

$$I^{\text{mark}} = \delta_{t_{\text{f}}}(\delta_{t_{\text{c}}}(I^{\text{blob}}, B_{\text{c}}), B_{\text{f}}) \tag{11}$$

However, in some cases I^{blob} may still contain boundary information. To

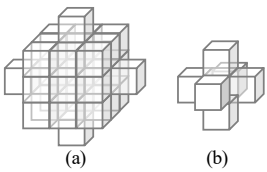

Fig. 16 (a) coarse 3D structuring element B_{c} and (b) fine 3D structuring element B_{f} for 3D conditional erosion

refine it we use 3D conditional erosion with a coarse structuring element B_c and a fine structuring B_f as shown in Figure 16. To achieve this we first identify the connected components [56] in I^{blob} and then iteratively erode each connected component using B_c until its size is smaller than some threshold t_c. We then continue eroding each connected component using B_f until its size is smaller than another threshold t_f. We denote the final result by I^{mark}, as shown in Equation 11, where $\delta_{t_c}(I^{blob}, B_c)$ defines iterative erosion of all connected component in I^{blob} by a coarse structuring element B_c until the size of each component is smaller than the coarse object threshold t_c. Similarly, $\delta_{t_f}(I^{blob}, B_f)$, denotes iterative erosion by a fine structuring element, B_f, until the size of the object is smaller than the fine object threshold t_f. Finally, marker-controlled watershed [39] is used to generate instance segmentation masks I^{seg} using I^{mark}. Small objects in I^{seg} that are less than 20 voxels in size are then removed, and each object color coded for visualization.

3.4 Comparison Methods

We compared NISNet3D with several deep learning image segmentation methods including VNet [27], 3D U-Net [26], Cellpose [3], DeepSynth [6], and StarDist3D [15]. In addition, we also compare NISNet3D with several commonly used biomedical image analysis tools including 3D Watershed [39], Squassh [40], CellProfiler [41], and VTEA [42].

VNet [27] and 3D U-Net [26] are two popular 3D encoder-decoder networks with shortcut concatenations designed for biomedical image segmentation. Cellpose [3] uses a modified 2D U-Net for estimating image segmentation and spatial flows, and uses a dynamic system [3] to cluster pixels and further separate touching nuclei. When segmenting 3D volumes, Cellpose works from three different directions slice by slice and combines the 2D segmentation results into a 3D segmentation volume [3]. DeepSynth uses a modified 3D U-Net to segment 3D microscopy volumes and uses watershed to separate touching nuclei [6]. Similarly, StarDist3D uses a modified 3D U-Net to estimate the star-convex polyhedra used to represent nuclei [15].

Alternative non deep-learning based techniques such as 3D Watershed [39] uses the watershed transformation [55] and conditional erosion [39] for to perform nuclei segmentation. Similarly, Squassh, an ImageJ plugin for both 2D and 3D microscopy image segmentation, uses active contours [40]. CellProfiler is an image processing toolbox and provides customized image processing and analysis modules [41], whereas VTEA is an ImageJ plugin that combines various approaches including Otsu's thresholding and watershed to segment 2D nuclei slice by slice and reconstruct the results into a 3D volume [42].

For comparison purposes, we trained and evaluated the methods above using the same dataset as used for NISNet3D. We also used the same training and evaluation strategies used for NISNet3D as described in Section 3.5. We provide a discussion of the results in Section 2.6. Note that 3D Watershed, Squassh, CellProfiler, and VTEA do not need to be trained because they use more traditional image analysis techniques.

Table 4 The training evaluation strategies for NISNet3D and comparison methods

Parameters For Generating Synthetic Microscopy Volumes							NISNet3D-slim Training Strategies		NISNet3D-slim
Volume Name	$a_{\mathbf{min}}$	$a_{\mathbf{max}}$	$t_{\mathbf{o}}$	N	d	σ	**Version**	**Training**	**Evaluation**
$\mathcal{V}_1$	4	8	5	500	10	1	$\mathcal{M}_1$	50 volumes of synthetic $\mathcal{V}_1$	Tested on 1 subvolume of $\mathcal{V}_1$
$\mathcal{V}_2$	10	14	10	70	4	1	$\mathcal{M}_2$	50 volumes of synthetic $\mathcal{V}_2$	Tested on 16 subvolumes of $\mathcal{V}_2$
$\mathcal{V}_3$	6	10	200	200	5	2	$\mathcal{M}_3$	50 volumes of synthetic $\mathcal{V}_3$	3-fold cross-validated on 9 subvolumes of $\mathcal{V}_3$
$\mathcal{V}_4$	8	10	100	560	4	4	$\mathcal{M}_4$	lightly retrained using $\mathcal{M}_3$ on 1 subvolume of $\mathcal{V}_4$	Tested on 4 subvolumes of $\mathcal{V}_4$
$\mathcal{V}_5$	-	-	-	-	-	-	$\mathcal{M}_5$	27 subvolumes of $\mathcal{V}_5$	Tested on 27 subvolumes of $\mathcal{V}_5$

3.5 Experimental Settings

The parameters used for generating I^{bi} are provided in Table 4 where ($a_{\min}$, $a_{\max}$) is the range of the ellipsoid semi-axes lengths, t_{ov} is the maximum allowed overlapping voxels between two nuclei, and N is the total number of nuclei in a synthetic volume. These parameters are based on visual inspection of nuclei characteristics in actual microscopy volumes.

SpCycleGAN was trained on unpaired I^{bi} and I^{orig} and the trained model was used to generate synthetic microscopy volumes. The weight coefficients of the loss functions were set to $\lambda_1 = \lambda_2 = 10$ based on the experiments described in [7]. The synthetic microscopy volumes were verified by a biologist (one of the co-authors).

For 3D Watershed, we used a 3D Gaussian filter to preprocess the images and used Otsu's method to segment/isolate objects from the background of each volume. Subsequently the 3D conditional erosion described in Section 3.3.2 was used to obtain the markers, and the marker-controlled 3D watershed method, which was implemented by the Python Scikit-image library, was used to separate touching nuclei.

In the case of CellProfiler [41], customized image processing modules including inhomogeneity correction, median filtering, and morphological erosion were used to preprocess the images, and the default "IdentifyPrimaryObject" module was used to obtain 2D segmentation masks for each slice. The 2D segmentations were then merged into a 3D segmentation volume using the blob-slice method described in [42, 57].

With regards to Squassh [40], we used "Background subtraction" with a manually tuned "rolling ball window size" parameter. The rest of the parameters were set to their default values. We observed that Squassh did fairly well on volume $\mathcal{V}_2$ but totally failed on $\mathcal{V}_4$ due to the densely clustered nuclei. For VTEA [42], a Gaussian filter and background subtraction were used to preprocess the image. The object building method was set to "Connect 3D", and the segmentation threshold determined automatically. We manually tuned the parameters "Centroid offset", "Min vol", and "Max vol" to obtain the best visual segmentation results. Finally, watershed was utilized to perform segmentation. Since VTEA and Cellprofier's "IdentifyPrimaryObject" modules only work on 2D images, we see that their segmentation results, shown in Figure 5, suffer from over-segmentation errors in the XZ planes.

As far as VNet [27], 3D U-Net [26] and DeepSynth [6] are concerned, we improved the segmentation results by using our 3D conditional erosion described in Section 3.3.2 followed by 3D marker-controlled watershed to separate touching nuclei. In the case of Cellpose, we used the "nuclei" style and since the training of Cellpose is only limited to 2D images, we trained Cellpose on every XY focal planes of our subvolumes following the training schemes given in Table 4. We observe that Cellpose has trouble capturing some very large or small nuclei in an input subvolume and performs worse on "thinner" subvolumes containing more non-ellipsoidal nuclei. Figure 7 shows the 2D to 3D reconstruction errors for Cellpose and VTEA when compared with

NISNet3D's results. For StarDist, we observed that it has difficulty segmenting concave objects or objects that are not ellipsoidal or star like in shape (non-star-convex objects) in $\mathcal{V}_3$ and achieves better performance on regular ellipsoidal nuclei in $\mathcal{V}_4$ (See Figure 5).

Both SpCycleGAN and NISNet3D were implemented using PyTorch. We used 9-block ResNet for the generators G, F , and the segmentor S (see Figure 11). The discriminators D_1 and D_2 (Figure 11) were implemented with the "PatchGAN" classifier [58]. SpCycleGAN was also trained with an Adam optimizer [59] for 200 epochs with an initial learning rate 0.0002 that linearly decays to 0 after the first 100 epochs. Figure 3 depicts the generated synthetic nuclei segmentation masks and corresponding synthetic microscopy images.

As mentioned above, we provide two versions of NISNet3D: NISNet3D-slim and NISNet3D-syn for different application scenarios. We trained 5 versions of NISNet3D-slim, denoted as $\mathcal{M}_1$-$\mathcal{M}_5$, using three training methods (See Table 4). First, we trained three models $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_3$ using the synthetic versions of the three volumes $\mathcal{V}_1$-$\mathcal{V}_3$, respectively. Next, we transfer the weights from $\mathcal{M}_3$ and continue training on a limited number of actual microscopy subvolumes of $\mathcal{V}_4$ to produce the fourth model $\mathcal{M}_4$. Finally, we directly train $\mathcal{M}_5$ on subvolumes of actual microscopy data $\mathcal{V}_5$ only. After training, we evaluate the models $\mathcal{M}_1$-$\mathcal{M}_5$ through two different evaluation schemes. In one method we directly test the models $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_4$, and $\mathcal{M}_5$ using all the subvolumes of the actual microscopy volumes $\mathcal{V}_1$, $\mathcal{V}_2$, $\mathcal{V}_4$, and $\mathcal{V}_5$, respectively, since they were not used for training. In the case of $\mathcal{M}_3$, we use 3-fold cross-validation to split the actual microscopy volume $\mathcal{V}_3$ into 9 subvolumes that are randomly divided into 3 equal sets and then iteratively update the model on one of the sets and test on other two sets. We used cross-validation to show the effectiveness of our method when the evaluation data is limited. Note that when lightly retraining $\mathcal{M}_3$ we update all its parameters while continue to train on actual microscopy volumes. The training and evaluation scheme for all models is provided in Table 4.

NISNet3D-synth, on the other hand is trained on 800 synthetic microscopy volumes that include the synthetic versions of $\mathcal{V}_1$-$\mathcal{V}_4$, and tested separately on each manually annotated original microscopy datasets described in Table 1. NISNet3D-synth is designed for the situation where no ground truth annotations are available. In contrast, NISNet3D-slim is used for the case where limited ground truth annotated volumes are available. In this case synthetic volumes are used for training and a small amount of actual ground truth data is for used for light retraining.

Table 5 Parameters used for NISNet3D nuclei instance segmentation

	NISNet3D-slim					**NISNet3D-synth**			
Parameters	$\mathcal{V}_1$	$\mathcal{V}_2$	$\mathcal{V}_3$	$\mathcal{V}_4$	$\mathcal{V}_5$	$\mathcal{V}_1$	$\mathcal{V}_2$	$\mathcal{V}_3$	$\mathcal{V}_4$
T_{m}	5	5	1	1	0	0	0	4	0
T_{c}	700	3000	2000	2000	2000	2000	2000	2000	2000
T_{f}	200	500	700	300	200	200	200	500	200

Both NISNet3D-slim and NISNet3D-synth were trained for 100 epochs using the Adam optimizer [59] with a constant learning rate of 0.001. The weight coefficients of the loss function were set to $\lambda_3 = 1$, $\lambda_4 = 10$, $\lambda_5 = 10$, the hyper-parameters β, γ of $\mathcal{L}_{\text{FL}}$ were set to 0.8 and 2, respectively, and the hyper-parameters α_1, α_2 in $\mathcal{L}_{\text{TL}}$ were set to 0.3 and 0.7 [54, 60], respectively. The nuclei instance segmentation parameters used in our experiments are provided in Table 5.

3.6 Evaluation Metrics

We use object-based metrics to evaluate nuclei instance segmentation accuracy. To this end, we define N^t_{TP} as the number of True Positive detections when the Intersection-over-Union (IoU) [61] between a detected nucleus and a ground truth nucleus is greater than some threshold of t voxels. Similarly, N^t_{FP} is the number of False Positives greater than t voxels, and N^t_{FN} is the number of False Negatives [62, 63] greater than t voxels, respectively. N^t_{TP} measures how many nuclei in a volume are correctly detected, and the higher the value of N^t_{TP} the more accurate the detection. N^t_{FP} represents the detected nuclei that are not actually nuclei but are false detections, and N^t_{FN} represents the number of nuclei that were not detected. A precise and accurate detection method should have high N^t_{TP} but low N^t_{FP} and N^t_{FN} values.

Based on N^t_{TP}, N^t_{FP}, N^t_{FN} we define the following metrics that are used to reduce bias [43]: mean Precision ($\text{mP} = \frac{1}{|T_{\text{IoUs}}|}\sum_{t\in T_{\text{IoUs}}} \frac{N^t_{\text{TP}}}{N^t_{\text{TP}}+N^t_{\text{FP}}}$), mean Recall ($\text{mR} = \frac{1}{|T_{\text{IoUs}}|}\sum_{t\in T_{\text{IoUs}}} \frac{N^t_{\text{TP}}}{N^t_{\text{TP}}+N^t_{\text{FN}}}$), and mean F_1 score ($\text{mF}_1 = \frac{1}{|T_{\text{IoUs}}|}\sum_{t\in T_{\text{IoUs}}} \frac{2N^t_{\text{TP}}}{2N^t_{\text{TP}}+N^t_{\text{FP}}+N^t_{\text{FN}}}$) that are the mean of Precision, Recall, and the F_1 score, respectively, over a set of multiple IoU thresholds T_{IoUs}. We set $T_{\text{IoUs}} = \{0.25, 0.3, ..., 0.45\}$ for datasets $\mathcal{V}_1$-$\mathcal{V}_4$, and set $T_{\text{IoUs}} = \{0.5, 0.55, ..., 0.75\}$ for datset $\mathcal{V}_5$. In addition, we obtained the Average Precision (AP) [44, 45] (a commonly used object detection metric), by estimating the area under the Precision-Recall Curve [46], using the same sets of thresholds in T_{IoUs}. For example, $\text{AP}_{.25}$ is the average precision evaluated at an IoU threshold of 0.25. The mean Average Precision (mAP) is then obtained as $\text{mAP} = \frac{1}{|T_{\text{IoUs}}|}\sum_{t\in T_{\text{IoUs}}} \text{AP}_t$.

The reason for using different thresholds T_{IoUs} for $\mathcal{V}_1$-$\mathcal{V}_4$ compared to those used for $\mathcal{V}_5$ is that throughout the experiments we observed that the nuclei in datasets $\mathcal{V}_1$-$\mathcal{V}_4$ are more challenging to segment than the nuclei in $\mathcal{V}_5$. Using the same IoU thresholds for evaluating all the datasets resulted in a lower evaluation accuracy for the volumes $\mathcal{V}_1$-$\mathcal{V}_4$ than for $\mathcal{V}_5$. Thus, we chose two different sets of IoU thresholds for $\mathcal{V}_1$-$\mathcal{V}_4$, and $\mathcal{V}_5$, respectively.

Finally, we use the Aggregated Jaccard Index (AJI) [47] to integrate object and voxel errors. The AJI is defined as:

$$\text{AJI} = \frac{\sum_{i=1}^{N} |G_i \cup S^i_m|}{\sum_{i=1}^{N} |G_i \cap S^i_m| + \sum_{j\in U} |S_j|} \tag{12}$$

where G_i denotes the ith nucleus in the ground truth volume G having a total number of N nuclei, U is the volume of segmented nuclei without corresponding ground truth, and S_m^i is the mth connected component in the segmentation mask which has the largest Jaccard Index/similarity with G_i. Note that each segmented nucleus with index m, that is belongs mth connected component , cannot be used more than once.

The values of all the parameters for each method were chosen to achieve the best visual results, as discussed in Section 2.6. The values of the various metrics used are given in Table 2 and Table 3 for each microscopy dataset. Figure 4 shows the AP scores using multiple IoU thresholds and the box plot of AJI for each subvolume of the dataset $\mathcal{V}_2$. The orthogonal views (XY focal planes and XZ focal planes) of the segmentation masks are overlaid on the original microscopy subvolume for each method on $\mathcal{V}_1$-$\mathcal{V}_5$ as shown in Figure 5 and Figure 6. Note that the different colors correspond to different nuclei.

3.7 Visualizing the Differences

Entire microscopy volumes contain many regions with varying spatial characteristics. In order to see how the segmentation methods perform on various regions, we propose three methods for visualizing the errors and the differences between a "test segmented volume" and a "reference segmented volume". We use the NISNet3D segmented volume as the reference segmented volume to visualize the segmentation differences between VTEA and NISNet3D, and also between DeepSynth and NISNet3D for the entire volumes. The three methods employed will be referred to as *Visualization Method A*, *B*, and *C*, respectively, and use an *Overlay Volume* and three types of *Difference Volumes*. Note that *Visualization Method A* does not need a "reference segmented volume" whereas *Visualization Method B* and *C* need a "reference segmented volume".

Next we describe how to generate an *Overlay Volume*. Using the notation shown in Figure 10, we denote an original microscopy volume as I^{orig} and m as the maximum intensity of I^{orig}. For a "reference segmented volume" (obtained via NISNet3D), we denote the binary segmentation masks for I^{orig} as I^{mask} and denote the color-coded segmentation for I^{orig} as I^{seg}. The R, G, and B components of I^{seg} are denoted by $I_{\text{R}}^{\text{seg}}$, $I_{\text{G}}^{\text{seg}}$, and $I_{\text{B}}^{\text{seg}}$, respectively. Similarly, for a "test segmented volume" (obtained via any other method), we denote C^{bi} as the binary segmentation masks for I^{orig}, and denote C as the color-coded segmentation of I^{orig} with R, G, and B components C_{R}, C_{G}, and C_{B}, respectively.

We denote the *Overlay Volume* as L having R, G, and B components L_{R}, L_{G}, and L_{B}, respectively. The *Overlay Volume* for a "test segmented volume" is generated by adding the original microscopy volume to each of the R, G, and B components of the color-coded segmented volume, as described in Equation 13.

$$L_{\text{R}} = I^{\text{orig}} + C_{\text{R}}, L_{\text{G}} = I^{\text{orig}} + C_{\text{G}}, L_{\text{B}} = I^{\text{orig}} + C_{\text{B}} \tag{13}$$

Let $S^{3D} = \{s_1^{3D}, s_2^{3D}, ..., s_n^{3D}\}$ be the set of all 3D segmented nuclei in C, where s_i^{3D} is a volume having the same size as C but only contains the ith segmented 3D nucleus, and let $S^{2D} = \{s_1^{2D}, s_2^{2D}, ..., s_k^{2D}\}$ be the set of all 2D objects in C from each XY focal plane, where s_i^{2D} is a volume of same size as C but only contains the ith segmented 2D nucleus. Similarly, let $O^{3D} = \{o_1^{3D}, o_2^{3D}, ..., o_s^{3D}\}$ be the set of all 3D objects in I^{seg}, and let $O^{2D} = \{o_1^{2D}, o_2^{2D}, ..., o_j^{2D}\}$ be the set of all 2D objects in I^{seg} from each slice. We use S^{3D} and O^{3D} in our generation of the three types of *Difference Volumes* using the *Visualization Methods A*, *B*, and *C*.

3.7.1 Visualization Method A

The *Difference Volume* generated by *Visualization Method A* shows voxels in the original microscopy volume that are not segmented by either test or reference methods. The input to *Visualization Method A* is the original microscopy volume and a segmented volume ("test segmented volume" or "reference segmented volume"). Using VTEA as an example: a VTEA segmented volume is weighted by the maximum intensity m, subtracted from the original microscopy volume, and negative differences set to 0 as indicated in Equation 14.

$$I^{A} = \max(I^{orig} - m * C^{bi}, 0) \tag{14}$$

This operation produces *Difference Volume*, I^{A}, which highlights the voxels in the original microscopy image that are not segmented. We replace C^{bi} in Equation 14 with I^{mask} to obtain the *Difference Volume* for the "reference segmented volume" and NISNet3D. Figure 8 ((a), (b), (c)) shows the *Difference Volumes* generated by *Method A* for VTEA, DeepSynth, and NISNet3D using dataset $\mathcal{V}_3$.

3.7.2 Visualization Method B

The *Difference Volume*, I^{B}, generated by *Visualization Method B* emphasizes the under-segmentation regions, that is the regions where multiple nuclei in the "reference segmented volume" are detected as a single nucleus in the "test segmented volume". Here we use NISNet3D as "reference segmented volume" and use VTEA or DeepSynth as "test segmented volume". The input to *Visualization Method B* is a VTEA (or DeepSynth) segmented volume and a NISNet3D segmented volume.

Using VTEA as an example: if two or more nuclei in the NISNet3D segmented volume intersect with the same single nucleus in the VTEA segmented volume, we emphasize the single nucleus segmented by VTEA in I^{B}, using Equation 15.

$$\begin{aligned} I^{B} = \bigcup \{ & s^{2D} \in S^{2D} : \exists o_p^{2D} \in O^{2D}, o_q^{2D} \in O^{2D}, o_p^{2D} \neq o_q^{2D} \\ & s^{2D} \cap o_p^{2D} \neq \emptyset, s^{2D} \cap o_q^{2D} \neq \emptyset \} \end{aligned} \tag{15}$$

The resulting volume I^{B} is overlaid on the original microscopy volume using Equation 13. Figure 9 depicts the *Difference Volumes* generated by *Visualization Method B* on a subvolume of dataset $\mathcal{V}_3$.

3.7.3 Visualization Method C

The *Difference Volume*, I^{C}, generated by *Visualization Method C* highlights nuclei segmented by a "reference segmented volume" but are completely missed by a "test segmented volume". The input to *Visualization Method C* is a VTEA (or DeepSynth) segmented volume and a NISNet3D segmented volume. Using VTEA as an example: if the voxels of a nucleus in NISNet3D segmented volume do not intersect with any voxels of any segmented nucleus from the VTEA segmented volume, then I^{C} will emphasize this nucleus from NISNet3D, using Equation 16:

$$I^{\mathrm{C}} = \bigcup \{o^{\mathrm{3D}} \in O^{\mathrm{2D}} : \neg\exists s^{\mathrm{2D}} \in S^{\mathrm{2D}}, s^{\mathrm{2D}} \cap s^{\mathrm{3D}} \neq \emptyset\} \tag{16}$$

The *Difference Volumes* for *Visualization Method A* and *C* are shown in Figure 8.

4 ACKNOWLEDGMENT

This research was partially supported by a George M. O'Brien Award from the National Institutes of Health under grant NIH/NIDDK P30 DK079312 and the endowment of the Charles William Harrison Distinguished Professorship at Purdue University.

The authors have no conflicts of interest.

The original image volumes used in this work were provided by Malgorzata Kamocka, Sherry Clendenon, and Michael Ferkowicz at Indiana University. We gratefully acknowledge their cooperation.

The NISNet3D source code package is available upon request to imart@ecn.purdue.edu. The source code is released under Creative Commons License Attribution-NonCommercial-ShareAlike - CC BY-NC-SA. The source code cannot be used for commercial purposes. The test volumes are also available at [37].

Address all correspondence to Edward J. Delp, ace@ecn.purdue.edu