Combining max-tree and CNN for segmentation of cellular FIB-SEM images

Cyril Meyer[1] [0000−0001−7262−999X] ⋆,
Étienne Baudrier[1] [0000−0002−2038−9434],
Patrick Schultz[2] [0000−0002−7310−6186], and
Benoît Naegel[1] [0000−0002−7695−1473]

[1] ICube, UMR 7357 CNRS, Université de Strasbourg
[2] IGBMC, UMR 7104 CNRS, U1258 INSERM, Université de Strasbourg

Abstract. Max-tree (or component-tree) is a hierarchical representation which associates to a scalar image a descriptive data structure induced by the inclusion relation between the binary components obtained at successive level-sets. Various attributes related to these binary components can be computed and stored into the tree.
Max-trees have been involved in many applications, enabling to perform attribute filtering in an efficient algorithmic way. Since the resulting images do not contain any new contour, these kind of filters are called *connected operators*.
In this paper, we propose to rely on max-trees and attribute filters to enrich the input of a convolutional neural network (CNN) to improve a task of segmentation. More precisely, two approaches are considered: a first approach in which images are preprocessed using attribute filters and a second approach in which maps of attributes relying on max-trees are computed. Based on these two different approaches, the resulting maps are used as additional input in a standard CNN in a context of semantic segmentation.
We propose to compare different attributes and nodes selection strategies and to experiment their usage on a practical problem: the segmentation of the mitochondria and endoplasmic-reticulum in Focused Ion Beam milling combined with Scanning Electron Microscopy (FIB-SEM) images.
We provide original images, annotations, source code and a documentation to reproduce the experimentation results.

Keywords: Mathematical Morphology · Connected Operators · Max-tree · Segmentation · Deep Learning · Convolutional Neural Network · Electron Microscopy · FIB-SEM.

1 Introduction

The max-tree structure [21,16] allows to perform efficiently attribute filtering [7] and has been involved in many image processing applications. The resulting operators are called *connected* [22] since they do not create new contours nor modify

⋆ IdEx Doctoral contract, Université de Strasbourg

their position. In [3], Farfan *et al.* have suggested that max-tree attributes could be used to feed a deep convolutional neural network (CNN) in order to improve the results of detection and segmentation tasks. Following and generalizing this approach, the aim of this paper is to provide a reproducible framework enabling to perform various experiments involving max-trees and CNN in a context of semantic segmentation of cellular FIB-SEM images.

Our contributions are twofold: in a first approach, input images are preprocessed using various attribute filters [12] and then concatenated as additional inputs of a CNN. In a second approach, maps of attributes are computed from the max-tree and then added in a CNN, following Farfan *et al.* approach.

Finally, our work aims to be handy. For this purpose, all the methods we propose can be used on a high-end workstation and do not require large GPU/TPU clusters. Also, our source code, datasets (original images, annotations) and documentation are publicly available, allowing everybody to reproduce the results, but also to reuse the code for their own needs.

2 State of the art

To address the segmentation of cellular electron microscopy images, the state-of-the-art methods are currently based on CNN [5,9,19,2,18,24,12] and the U-Net architecture remains mainly used. However, despite the good accuracy that can be obtained using these methods, the resulting segmentations can still suffer from various imperfections. In particular, thin and elongated objects such as endoplasmic reticulum can be disconnected and some parts may be distorted [12]. These effects may be the result of the context window that is fixed and too narrow in the first layer of the CNN, preventing to capture sufficient global information.

To overcome this, Farfan *et al.* [3] have proposed to enrich a CNN with attributes computed from the max-tree, enabling to capture at a pixel level, an information that may be non-local.

In the sequel of this paper, we will explore various strategies in order to incorporate max-tree attributes into CNN with the aim of potentially improving segmentation results.

3 Methods

3.1 Max-tree

Let $I : E \to V$ be a discrete, scalar (i.e. grayscale) image, with $E \subseteq \mathbb{Z}^n$ and $V \subseteq \mathbb{Z}$. A cut of I at level v is defined as: $X_v(I) = \{p \in E | I(p) \geq v\}$. Let $C[X]$ be the set of connected components of X. Let Ψ be the set of all the connected components of the cuts of I:

$$\Psi(I) = \bigcup_{v \in V} \{C[X_v(I)]\}$$

The relation $\subseteq$ is a partial order on Ψ. The transitive reduction of the relation $\subseteq$ on Ψ induces a graph called the Hasse diagram of $(\Psi, \subseteq)$. This graph is a tree, the root of which is E. The rooted tree $\mathcal{T} = (\Psi, L, E)$ is called the max-tree of I, with Ψ, L, E being respectively the set of nodes, the set of edges and the root of $\mathcal{T}$. The parent node of N, denoted $Par(N)$, is the unique node such that: $(Par(N), N) \in L$ and $N \subseteq Par(N)$ for $N \neq E$. The *branch* associated to a node is the set of its ancestors and is defined for a node $N \in \Psi$ by: $Br(N) = \{X \in \Psi \mid X \supseteq N\}$.

In this work, the computation of the max-tree is based on the recursive implementation of Salembier [21], and node attributes are computed during the construction of the tree. In the rest of this paper, we will focus on the following attributes which have been proposed in the literature:

- The height H is the minimum gray level of the connected component [21].
- The area A is the number of pixels in the connected component [21].
- The contour CT represents the number of pixels that have both a neighbor inside and outside the component [20].
- The contrast C is the difference between the max and minimum in the connected component [21].
- The complexity CPL represents the contour length $|CT|$ divided by the area [20].
- The compacity (sometimes named compactness or circularity) CPA is the area divided by the square of the contour length $|CT|^2$ [22].
- The volume V is the sum of the difference between the pixels values in the node and the node height [16].
- The mean gradient border MGB represents the mean value of the gradient magnitude for the contour pixels [3].

The tree attributes can be merged in order to compute an image, by associating to each pixel an attribute value computed from its corresponding nodes [3]. Each pixel p belongs to several nodes: the connected component N including p in the level-set $X_{I(p)}(I)$ and all the nodes belonging to its *branch* $Br(N)$. To associate a unique value to each pixel, different policies can be implemented: for example, by keeping the maximum, the minimum or the mean value of the attributes of the branch nodes [3].

In this work, we propose the following strategy. For each pixel p, the set of nodes belonging to the branch of p is retrieved, and only a subset of nodes having an attribute value in a certain range (given as a parameter) is kept. From this set, the node N_{best} optimizing a certain stability criterion is kept. Finally, the value of p in the resulting image is set to the attribute value of N_{best}. The resulting image is normalized in the range $V = [\![0, 255]\!]$.

The criterion used to retrieve the optimal node is based on the concept of Maximally Stable Extremal Regions proposed by Matas *et al.* [10]. The idea is to retrieve the most stable regions based on the area variation between successive nodes since these regions represent salient objects of the image. For each node $N \in \Psi$, with $N \neq E$ (i.e. different from the root), we define two stability attributes as follows:

$$\nabla_A(N) = \frac{|\ A(Par(N)) - A(N)\ |}{|H(Par(N)) - H(N)|} \cdot \frac{1}{A(N)}$$

$$\Delta_A(N) = |\nabla_A(Par(N)) - \nabla_A(N)|$$

where $Par(N)$ defines the parent node of N.

3.2 Segmentation

We base our segmentation method on fully convolutional networks for semantic segmentation. We use attribute filtered images or max-tree attribute maps as additional input of our network. We feed the network with the original and the preprocessed images, concatenating them in the color (spectral) channels.

For model architectures, we use a 2D U-Net [19], which is a reference for biomedical image segmentation. A 3D U-Net [2] could also be used, but the results are not necessarily better [23,17,25] and the computational cost of training the model is much higher. In a preliminary experiment, we have compared 2D and 3D models, using an equal number of parameters and same input size and the 2D U-Net perform equal, if not better as the 3D one.

Each block of the network is composed of convolutions with ReLU activation, followed by batch normalization [6] and a residual connection [4], see Figure 1. We use a 50% dropout entering the deepest block to avoid over fitting. We always use padded convolution to maintain the spatial dimension of the output. The model starts with 64 filters at the first level, for a total of 32.4 millions parameters.

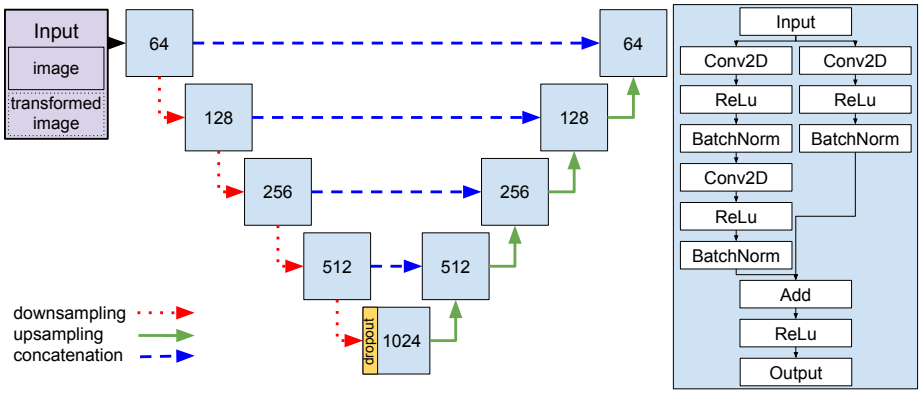


Fig. 1. U-Net architecture and its backbone block. The number in the box corresponds to the number of filters at this level for the convolutions blocks. The network is fed with the original and the preprocessed images, concatenating them in the color channels.

We train our models by minimizing the Dice loss. For the binary segmentation task, we use classical Dice loss and for multi-class segmentation problems, we use a weighted mean of the loss for each class, with the same weight ($W = 0.5$) for our two classes. We note X the ground truth, Y the prediction, W the weight list and C the classe list. The ε term is used for stability when $\sum (X + Y) = 0$ and is set to 10^{-4}.

$$L_{Dice}(X, Y) = 1 - \frac{2 \cdot \sum X \cdot Y + \varepsilon}{\sum (X + Y) + \varepsilon}$$

$$L_{DiceMean}(X, Y, W) = \frac{1}{|C|} \sum_{c \in C} W_c \cdot L_{Dice}(X_c, Y_c)$$

The model is trained for 128 epochs, each epoch is composed of 512 batches, a batch is composed of 8 patches and a patch is a $256 \times 256 \times C$ subpart of the image, with C the number of channels. We use random 90° rotation, horizontal and vertical flips for data augmentation on the patches. We trained our model using Adam [8] optimizer with the following parameters: $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 1\text{e}{-07}$.

3.3 Evaluation Metrics

To evaluate our models, we binarize the model prediction with a threshold at 0.5. To predict a slice, we use the whole slice to avoid the negative border effects of padding.

To evaluate our results, we use the F1-Score which is a region-based metric, and the average symmetric surface distance (ASSD) which is a boundary-based metrics. In fact, the F1-Score is equivalent to the Dice Score. We note respectively TP, FP and FN the cardinal of the sets of true positives, false positives and false negatives. X is the ground truth, Y the binary prediction and ∂X the boundary of X.

$$\text{F1-Score} = \frac{2 \times TP}{2 \times TP + FP + FN}$$

$$\text{ASSD}(X, Y) = \frac{\sum_{x \in \partial X} d(x, Y) + \sum_{y \in \partial Y} d(y, X)}{|\partial X| + |\partial Y|}$$

with $d(x, A) = \min_{y \in A} \|x - y\|_2$.

4 Experiments

In this section, we test the improvement of the segmentation thanks to the addition of filtered images in the input. We compare the original image input with the enriched version. We also compare in the same time a multi-class segmentation model and two binary segmentations models. Finally, we repeat each of these configurations 11 times. In total, 363 models have been trained for this

experiment, each training lasts about 5 hours with an NVIDIA GeForce RTX 2080 Ti.

First, we define the filters we use as our experiment variable. Attributes are selected for their potential help for the segmentation, we list the selected filters in Table 1 (attribute maps strategy) and 2 (connected operators strategy), and renamed them for sake of simplicity.

Table 1. List of used attribute maps filters and their names.

Name	Attribute	Criterion	Limit
Contrast$_{\Delta_A}$	C	Δ_A	$0 \leq A \leq 1024^2$
Complexity$_{\Delta_A}$	CPL	Δ_A	$0 \leq A \leq 1024^2$
Compacity$_{\Delta_A}$	CPA	Δ_A	$CPA \leq 50$
Volume$_{\Delta_A}$	V	Δ_A	$0 \leq A \leq 1024^2$
MGB	MGB	MGB	$0 \leq A \leq 1024^2$

Table 2. List of used connected filters and their names. The "inverse" column is checked when filter is applied on the inverted image.

Name	Attribute	Range	Inverse
Contrast α	C	$[10, 150]$	
Contrast β	C	$[10, 150]$	✓
Area α	A	$[2 \times 32^2, 1024^2]$	
Area β	A	$[64^2, +\infty]$	
Area γ	A	$[16^2, 512^2]$	✓

Before processing the image with a max-tree, we apply a low pass filter (9×9 mean filter).

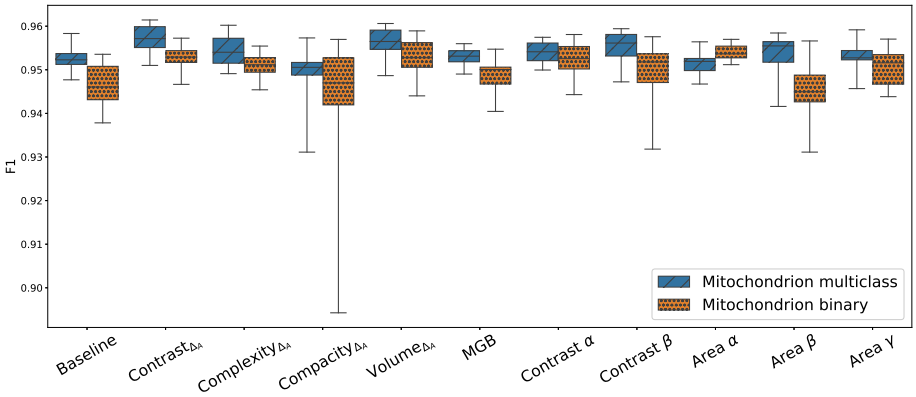
4.1 Data

We perform our experimentation on a stack of 80 slices from a 3D FIB-SEM image. Each slice has a size of 1536×1408. The image represents a HeLa cell and has an $(x \times y \times z)$ resolution of 5 nm $\times$ 5 nm $\times$ 20 nm. A ground truth is available on the stack for two kinds of organelles (i.e. cell subunits): mitochondria and endoplasmic reticulum. A default background class is affected to non-assigned pixel. An example slice with label is available in Figures 3 and 4 in Section A.2. Figures 5 to 14 depict the slice for each applied filter.

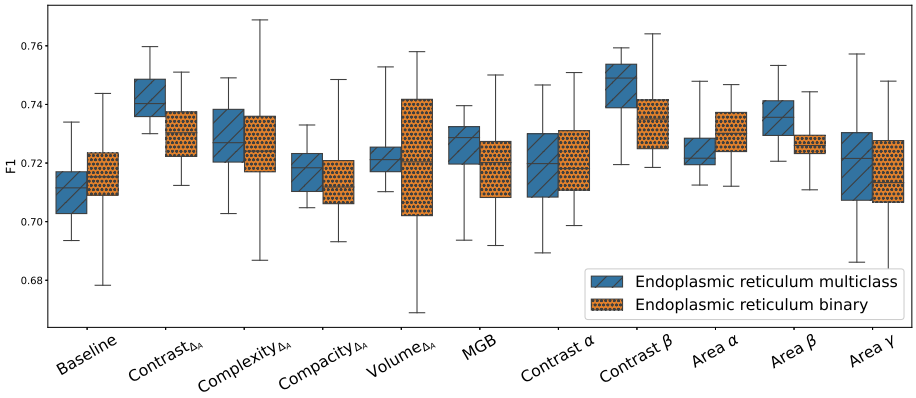
We divide the stack into 3 sets: training (first 40 slices), validation (next 20 slices) and test (last 20 slices). The training set is used to train the network, the validation set to select the best model during the training and the test set to provide evaluation metrics.

4.2 Results

Figures 2a and 2b depict the F1-score on the two classes of segmentation as box plots. Detailed mean and variation score are available in Table 3 and 4 in Section A.1.



(a) F1-score on mitochondria.



(b) F1-score on endoplasmic reticulum.

Fig. 2. The box shows the quartiles of the results, while the whiskers extend to show the rest of the distribution.

Baseline segmentation results Mitochondria are well segmented with a median F1-score up to 95% in multi-class segmentation and 94% in binary segmentation, which let a little possible improvement. Median F1-scores for reticulum up to 72% and 71% in multi-class and binary segmentation respectively. This thin organelle is indeed more difficult to segment.

Additional-input segmentation results On the mitochondria, the additional inputs improve the results for 11 cases out of 20 and 17 out of 20 for the reticulum. The gain on the reticulum is very interesting since it is the more difficult to segment for the baseline setup. The following additional inputs improve the result in all the four tests (binary and multi-class segmentations on mitochondria and reticulum): Contrast$_{\Delta_A}$, Complexity$_{\Delta_A}$, Contrast β. Moreover, the whiskers show that only Contrast$_{\Delta_A}$and Contrast β have a good stability in this experiment. These two inputs are therefore good candidates to be additional inputs to improve segmentation results. On the contrary, Compacity$_{\Delta_A}$and MGB do not yield to any improvement.

4.3 Reproducibility

In this section, we present the required steps to reproduce the results we presented.

We follow the ACM definition of reproducibility: *"The measurement can be obtained with stated precision by a different team using the same measurement procedure, the same measuring system, under the same operating conditions, in the same or a different location on multiple trials. For computational experiments, this means that an independent group can obtain the same result using the author's own artifacts."* [1]
For this purpose, our codes and datasets are publicly available. The project is split between three subprojects. First, the experimentation detailed documentation, training scripts, evaluation scripts, preprocessing script, results logs [11]. Second, the max-tree related functions and preprocessing script [14]. Finally, the dataset with images and annotations [13]
The following information are also available on the documentation repository with more details.

Requirements A system with Ubuntu 18.04.6 (or compatible) with `g++` and `git` installed. Python 3.6.9 with an environment including TensorFlow 2.6.2, NumPy, SciPy, scikit-image and MedPy.

Image preprocessing

- Prepare data for extraction with low pass filter.
  `python 01_mean_filter.py`
- Extract attribute image from pre-processed images.
  `./build_bin_requirements.sh`
  `./02_attribute_image.sh`
- Crop the image to the annotated area and construct a tiff stack.
  `python 03_crop_roi.py`

Network training and evaluation For the following commands, `$ID` is a unique identifier for the train, `$INPUT` is the folder containing the dataset, `$OUTPUT` is the folder containing the trained models and evaluation metrics, `$DATASET` select

the dataset to use (in our case, binary or multi-class dataset), `$SETUP` select the experiment to run. An automation bash script is available on the repository to run all the 33 setups once.

- Train the networks
  `python train.py $ID $INPUT $OUTPUT $DATASET $SETUP`
- Evaluate the networks
  `python eval.py $ID $INPUT $OUTPUT $DATASET $SETUP BEST`

Result analysis and figures reproduction Since the output of each model evaluation is a comma separated values (CSV) file, the analysis of the results can be done using various tools. We propose to use a Jupyter notebook with Pandas and Seaborn, merging the CSV files into a single dataframe. An example `analysis.ipynb` notebook is provided on the GitHub, which we use to produce our result figures and tables.

5 Conclusion

In this paper, we have presented a detailed experimental setup to evaluate the use of additional input in a CNN based segmentation task. The additional inputs are attribute maps obtained from a max-tree representation of the image. The evaluation is made on segmentation tasks in the context of 3D electronic microscopy. If most of the additional inputs improve one segmentation task, two of them – namely Contrast$_{\Delta_A}$ and Contrast β– improve all the tested segmentation tasks in terms of median F1-score and stability. Further than the segmentation results, the setup inspired from [3] has been entirely implemented in C++ and Python and is proposed in open access to make it reproducible.

As a perspective of this work, the feature extraction method based on max-tree attributes presented in this paper could be used for other applications. For example, it would be interesting to compute the max-tree directly inside the model and to use the attributes images as a nonlinear filter. Also, the attribute maps could be used as feature maps for more simple and explainable classifier as random forests or even decision tree. Besides, the Δ_A attributes we defined could be used in an interactive segmentation setup, where a single pixel will allow to select an interesting node, selecting an object connected component over a background. Finally, we proposed here a max-tree based method, but an extension to the tree of shapes [15] could be interesting and add more information to the image.

Acknowledgements We acknowledge the High Performance Computing Center of the University of Strasbourg for supporting this work by providing scientific support and access to computing resources. Part of the computing resources were funded by the Equipex Equip@Meso project (Programme Investissements d'Avenir) and the CPER Alsacalcul/Big Data. We thank D. Spehner from the Institut de Génétique et de Biologie Moléculaire et Cellulaire for providing the

images and V. Mallouh for providing the annotations. We acknowledge the use of resources of the French Infrastructure for Integrated Structural Biology FRISBI ANR-10-INBS-05 and of Instruct-ERIC. We acknowledge that this work is supported by an IdEx doctoral contract, Université de Strasbourg.

References

1. ACM: Artifact Review and Badging - V.1.1. https://www.acm.org/publications/policies/artifact-review-and-badging-current
2. Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In: Ourselin, S., Joskowicz, L., Sabuncu, M.R., Unal, G., Wells, W. (eds.) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016. pp. 424–432. Lecture Notes in Computer Science, Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-46723-8_49
3. Farfan Cabrera, D.L., Gogin, N., Morland, D., Naegel, B., Papathanassiou, D., Passat, N.: Segmentation of Axillary and Supraclavicular Tumoral Lymph Nodes in PET/CT: A Hybrid CNN/Component-Tree Approach. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 6672–6679 (Jan 2021). https://doi.org/10.1109/ICPR48806.2021.9412343
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778. IEEE, Las Vegas, NV, USA (Jun 2016). https://doi.org/10.1109/CVPR.2016.90
5. Heinrich, L., Bennett, D., Ackerman, D., Park, W., Bogovic, J., Eckstein, N., Petruncio, A., Clements, J., Pang, S., Xu, C.S., Funke, J., Korff, W., Hess, H.F., Lippincott-Schwartz, J., Saalfeld, S., Weigel, A.V.: Whole-cell organelle segmentation in volume electron microscopy. Nature pp. 1–6 (Oct 2021). https://doi.org/10.1038/s41586-021-03977-3
6. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: Proceedings of the 32nd International Conference on Machine Learning. pp. 448–456. PMLR (Jun 2015)
7. Jones, R.: Connected Filtering and Segmentation Using Component Trees. Computer Vision and Image Understanding **75**(3), 215–228 (Sep 1999). https://doi.org/10.1006/cviu.1999.0777
8. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)
9. Liu, J., Li, L., Yang, Y., Hong, B., Chen, X., Xie, Q., Han, H.: Automatic Reconstruction of Mitochondria and Endoplasmic Reticulum in Electron Microscopy Volumes by Deep Learning. Front. Neurosci. **14** (2020). https://doi.org/10.3389/fnins.2020.00599
10. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing **22**(10), 761–767 (Sep 2004). https://doi.org/10.1016/j.imavis.2004.02.006
11. Meyer, C.: CTAISegmentationCNN. GitHub repository (Oct 2022), https://github.com/Cyril-Meyer/DGMM2022-RRPR-CTAISegmentationCNN

12. Meyer, C., Mallouh, V., Spehner, D., Baudrier, É., Schultz, P., Naegel, B.: Automatic Multi Class Organelle Segmentation For Cellular Fib-Sem Images. In: 2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI). pp. 668–672 (Apr 2021). https://doi.org/10.1109/ISBI48211.2021.9434075
13. Meyer, C., Mallouh, V., Spehner, D., Schultz, P.: DGMM2022-RRPR-MEYER-DATA. GitHub repository (Oct 2022), https://github.com/Cyril-Meyer/DGMM2022-RRPR-MEYER-DATA
14. Meyer, C., Naegel, B.: ComponentTreeAttributeImage. GitHub repository (Oct 2022), https://github.com/Cyril-Meyer/DGMM2022-RRPR-ComponentTreeAttributeImage
15. Monasse, P., Guichard, F.: Fast computation of a contrast-invariant image representation. IEEE Transactions on Image Processing **9**(5), 860–872 (May 2000). https://doi.org/10.1109/83.841532
16. Najman, L., Couprie, M.: Building the Component Tree in Quasi-Linear Time. IEEE Transactions on Image Processing **15**(11), 3531–3539 (Nov 2006). https://doi.org/10.1109/TIP.2006.877518
17. Nemoto, T., Futakami, N., Yagi, M., Kumabe, A., Takeda, A., Kunieda, E., Shigematsu, N.: Efficacy evaluation of 2D, 3D U-Net semantic segmentation and atlas-based segmentation of normal lungs excluding the trachea and main bronchi. Journal of Radiation Research **61**(2), 257–264 (Mar 2020). https://doi.org/10.1093/jrr/rrz086
18. Oztel, I., Yolcu, G., Ersoy, I., White, T., Bunyak, F.: Mitochondria segmentation in electron microscopy volumes using deep convolutional neural network. In: 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). pp. 1195–1200 (Nov 2017). https://doi.org/10.1109/BIBM.2017.8217827
19. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. pp. 234–241. Lecture Notes in Computer Science, Springer International Publishing, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
20. Salembier, P., Brigger, P., Casas, J., Pardas, M.: Morphological operators for image and video compression. IEEE Transactions on Image Processing **5**(6), 881–898 (Jun 1996). https://doi.org/10.1109/83.503906
21. Salembier, P., Oliveras, A., Garrido, L.: Antiextensive connected operators for image and sequence processing. IEEE Transactions on Image Processing **7**(4), 555–570 (Apr 1998). https://doi.org/10.1109/83.663500
22. Salembier, P., Wilkinson, M.H.: Connected operators. IEEE Signal Processing Magazine **26**(6), 136–157 (Nov 2009). https://doi.org/10.1109/MSP.2009.934154
23. Srikrishna, M., Heckemann, R.A., Pereira, J.B., Volpe, G., Zettergren, A., Kern, S., Westman, E., Skoog, I., Schöll, M.: Comparison of Two-Dimensional- and Three-Dimensional-Based U-Net Architectures for Brain Tissue Classification in One-Dimensional Brain CT. Frontiers in Computational Neuroscience **15** (2022)
24. Xiao, C., Chen, X., Li, W., Li, L., Wang, L., Xie, Q., Han, H.: Automatic Mitochondria Segmentation for EM Data Using a 3D Supervised Convolutional Network. Front. Neuroanat. **12** (2018). https://doi.org/10.3389/fnana.2018.00092
25. Zettler, N., Mastmeyer, A.: Comparison of 2D vs. 3D Unet Organ Segmentation in abdominal 3D CT images. In: WSCG'2021 - 29. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2021 (2021). https://doi.org/10.24132/CSRN.2021.3101.5

A Appendix

A.1 Results

In the following tables, the mean and deviation are computed over the 8 best model out of 11, selected using the F1-score on the validation set.

Table 3. F1 score and ASSD for mitochondria segmentation

	Binary		Multi-class	
Setup	F1	ASSD	F1	ASSD
Baseline	0.949 ±0.004	3.742 ±1.970	0.952 ±0.003	5.761 ±5.315
Contrast$_{\Delta_A}$	**0.953** ±0.002	**1.208** ±0.616	**0.958** ±0.003	1.531 ±0.490
Complexity$_{\Delta_A}$	0.951 ±0.003	1.559 ±0.647	0.956 ±0.003	**1.211** ±0.384
Compacity$_{\Delta_A}$	0.949 ±0.006	2.495 ±3.027	0.951 ±0.001	2.786 ±2.305
Volume$_{\Delta_A}$	**0.954** ±0.003	1.226 ±0.260	**0.957** ±0.003	**1.280** ±0.370
MGB	0.950 ±0.003	2.124 ±0.996	0.953 ±0.002	1.654 ±0.345
Contrast α	**0.954** ±0.004	**1.007** ±0.274	0.955 ±0.002	1.639 ±0.932
Contrast β	0.951 ±0.006	1.797 ±0.873	**0.957** ±0.003	1.698 ±1.301
Area α	**0.954** ±0.002	1.499 ±0.839	0.952 ±0.003	1.950 ±0.916
Area β	0.948 ±0.005	2.459 ±2.478	0.955 ±0.002	1.711 ±0.492
Area γ	0.952 ±0.003	1.244 ±0.318	0.954 ±0.004	2.639 ±3.522

Table 4. F1 score and ASSD for endoplasmic reticulum segmentation

	Binary		Multi-class	
Setup	F1	ASSD	F1	ASSD
Baseline	0.721 ±0.013	7.758 ±0.740	0.718 ±0.013	8.282 ±1.037
Contrast$_{\Delta_A}$	0.738 ±0.009	7.767 ±0.877	**0.747** ±0.011	7.903 ±0.425
Complexity$_{\Delta_A}$	**0.741** ±0.015	7.720 ±0.689	0.735 ±0.011	**7.232** ±0.817
Compacity$_{\Delta_A}$	0.721 ±0.014	8.810 ±1.012	0.725 ±0.008	8.535 ±0.691
Volume$_{\Delta_A}$	**0.744** ±0.011	7.522 ±0.494	0.732 ±0.011	7.817 ±0.828
MGB	0.729 ±0.013	8.526 ±0.646	0.729 ±0.009	8.614 ±0.468
Contrast α	0.736 ±0.012	8.910 ±0.680	0.728 ±0.013	8.327 ±0.783
Contrast β	**0.741** ±0.016	7.715 ±0.520	**0.748** ±0.007	7.407 ±0.784
Area α	0.734 ±0.009	**6.414** ±0.677	0.728 ±0.011	7.448 ±1.507
Area β	0.733 ±0.006	8.296 ±1.494	0.741 ±0.007	7.862 ±0.901
Area γ	0.733 ±0.009	8.243 ±0.584	0.729 ±0.014	8.693 ±0.518

A.2 Example preprocessing visualization

Fig. 3. Original image

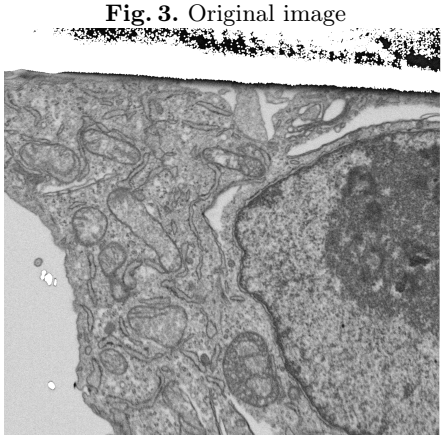


Fig. 4. Label, mitochondria in gray, endoplasmic reticulum in white

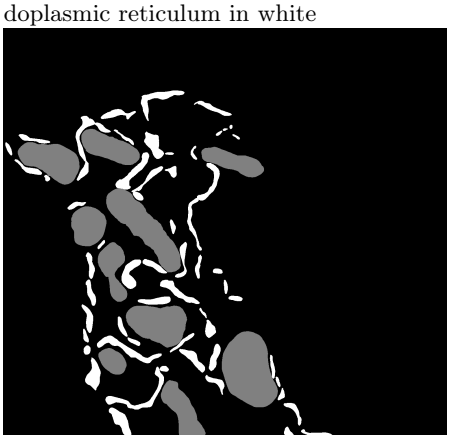


Fig. 5. Contrast$_{\Delta_A}$

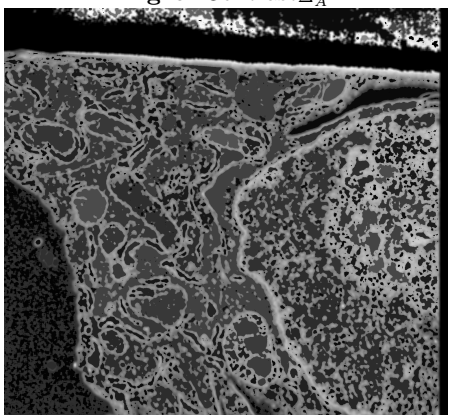


Fig. 6. Complexity$_{\Delta_A}$

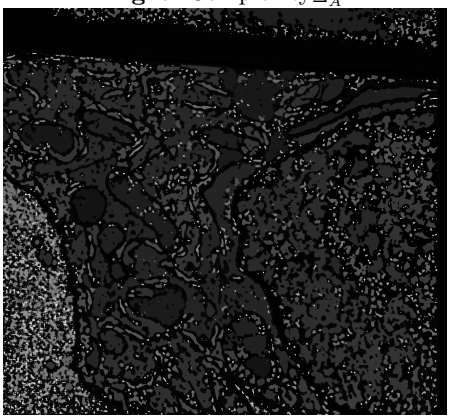


Fig. 7. Compacity$_{\Delta_A}$

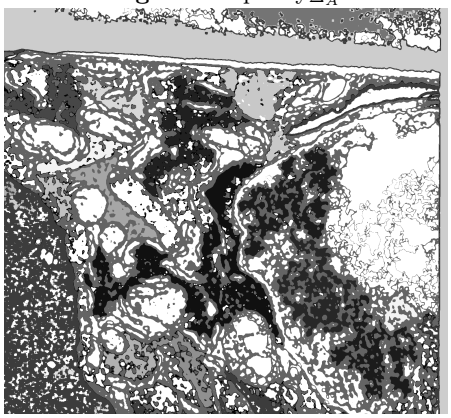


Fig. 8. Volume$_{\Delta_A}$

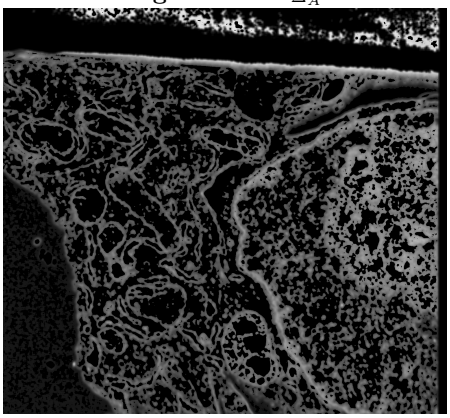


Fig. 9. MGB

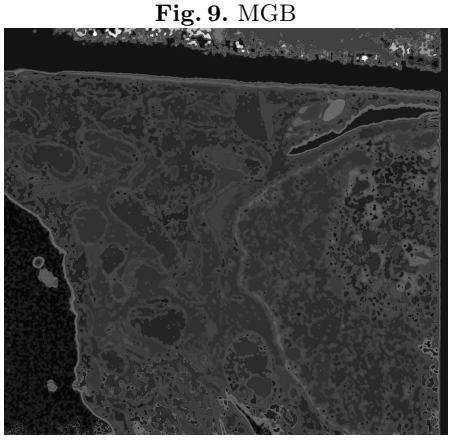


Fig. 10. Contrast α

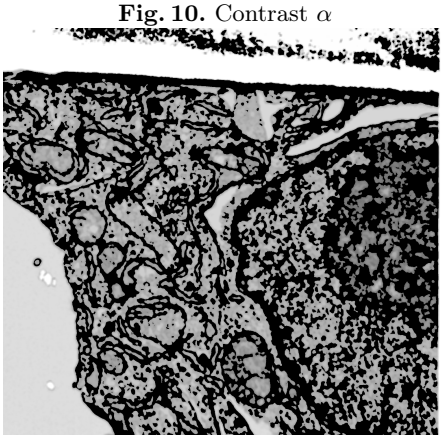


Fig. 11. Contrast β

Fig. 12. Area α

Fig. 13. Area β

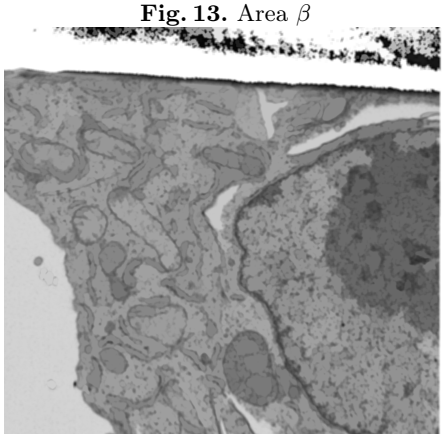


Fig. 14. Area γ