Label-free cell segmentation of diverse lymphoid tissues in 2D and 3D

Graphical abstract



Authors

John W. Wills, Jack Robertson, Pani Tourlomousis, ..., Huw D. Summers, Jonathan J. Powell, Paul Rees

Correspondence

jw2020@cam.ac.uk (J.W.W.), jjp37@cam.ac.uk (J.J.P.), p.rees@swansea.ac.uk (P.R.)

In brief

Cell segmentation is an essential step in quantitative tissue microscopy. Wills et al. show this can be achieved simply using the reflected laser light always present during routine imaging by confocal microscopy. This frees up microscope channels and establishes single-cell information as an attainable start point for many tissue microscopy experiments.

Highlights

- Cell segmentation of tissues can be achieved from reflected laser excitation light
- Single-cell information is freely established for many tissue microscopy studies
- Windows software is provided alongside extensive video tutorials and data





Report

Label-free cell segmentation of diverse lymphoid tissues in 2D and 3D

John W. Wills,^{1,2,4,*} Jack Robertson,¹ Pani Tourlomousis,¹ Clare M.C. Gillis,¹ Claire M. Barnes,² Michelle Miniter,¹ Rachel E. Hewitt,¹ Clare E. Bryant,¹ Huw D. Summers,² Jonathan J. Powell,^{1,*} and Paul Rees^{2,3,*}

¹Department of Veterinary Medicine, Cambridge University, Madingley Road, Cambridge CB3 0ES, UK

²Department of Biomedical Engineering, Swansea University, Fabian Way, Crymlyn Burrows, Swansea SA1 8EN, Wales, UK

³Imaging Platform, Broad Institute of MIT and Harvard, 415 Main Street, Boston, Cambridge, MA 02142, USA

⁴Lead contact

*Correspondence: jw2020@cam.ac.uk (J.W.W.), jjp37@cam.ac.uk (J.J.P.), p.rees@swansea.ac.uk (P.R.) https://doi.org/10.1016/j.crmeth.2023.100398

MOTIVATION Across the biomedical sciences, there is an urgent need to move beyond qualitative imaging to quantitative, cell-based reporting of tissue microscopy data. Typically, cell segmentation requires fluorescent labeling of nucleus and cytoplasm, which limits the spectral bandwidth available for other reporter molecules. However, recent advances in deep-learning algorithms have transformed automated image classification, and this raises the possibility of proceeding with reduced image information. Here, we show that 2D and 3D cell segmentation of lymphoid tissues can be freely established from the reflected laser excitation light always present during routine confocal microscopy using entirely standard equipment.

SUMMARY

Unlocking and quantifying fundamental biological processes through tissue microscopy requires accurate, *in situ* segmentation of all cells imaged. Currently, achieving this is complex and requires exogenous fluorescent labels that occupy significant spectral bandwidth, increasing the duration and complexity of imaging experiments while limiting the number of channels remaining to address the study's objectives. We demonstrate that the excitation light reflected during routine confocal microscopy contains sufficient information to achieve accurate, label-free cell segmentation in 2D and 3D. This is achieved using a simple convolutional neural network trained to predict the probability that reflected light pixels belong to either nucleus, cytoskeleton, or background classifications. We demonstrate the approach across diverse lymphoid tissues and provide video tutorials demonstrating deployment in Python and MATLAB or via standalone software for Windows.

INTRODUCTION

The analysis of tissues using fluorescence labeling and confocal microscopy represents a mainstay biomedical technique that is used worldwide to understand the biology of cells *in situ*.^{1–5} However, despite the accessibility of confocal microscopy and its ability to provide sensitive, quantifiable data with subcellular resolution in both 2D and 3D, the number of channels that can be successfully imaged is often limited in practice.^{2,6} Moving beyond qualitative observations to cell-based quantifications for every cell in a tissue specimen requires fluorescent staining (e.g., nuclei, cell membrane, or cluster of differentiation [CD] markers) to enable cell segmentation.^{2–5} However, these stains occupy channels that are often needed to fully address the bioclinical question.^{2,3,5} At the same time, as the number of fluorescence channels increases, so does the complexity, time requirement, and potential for channel cross-talk.^{5,7} Correcting this complicates

analysis and downstream data processing, increasing the expertise required and the risk of error. $^{3,5,7,8}_{\ }$

Recognizing these complexities in addition to the need to avoid phototoxicity and temporal errors during live-cell experiments with monolayer cells *in vitro*, microscopy techniques that harness endogenous contrast (i.e., label free) have been developed (e.g., phase/differential interference contrast, etc.).^{8–11} However, a recognized difficulty for subsequent, cell-based image analysis is that label-free cell segmentation accuracy decreases as cultures become confluent and cell-to-cell contact is established.^{9,10} In this regard, tissue environments are inherently complex and challenging as they are almost entirely comprised of contacting cells in 3D, layer-upon-layer arrangements.^{2,3,5}

An often overlooked capability of nearly all laser scanning confocal microscopes is the ability to capture reflected laser excitation light. Importantly, unlike transmitted light, this labelfree signal is filtered by the confocal aperture enabling capture



Cell Reports Methods Report

as 3D "z stacks" of optically isolated sections that can be simultaneously acquired alongside fluorescence information.^{7,8} Using diverse lymphoid tissues as an exemplar, here we demonstrate how this signal can be harnessed to provide accurate, labelfree cell segmentation in 2D and 3D. We show the approach can be deployed in conjunction with the user-friendly, opensource CellProfiler software,¹² enabling single-cell data extraction for image-based cell profiling in addition to reproducible workflow dissemination. The approach provides every cell and nearest-cell neighbor relationship *in situ* with high precision, leaving a spectrally unencumbered landscape for subsequent interrogation. To support uptake, we provide extensive video tutorials and data, demonstrating deployment using Python, MATLAB, or standalone software for Windows.

RESULTS

Figure 1 shows our strategy using mouse splenic tissue. Training data, exemplifying the cellular structure of the tissue, are collected from parallel tissue sections using simple, antibodyindependent fluorescent staining for cell nuclei and cytoskeletal F-actin (Figures 1A and 1B). During imaging, both fluorescence data and the backscattered reflectance signal from one of the excitation lasers are captured (Figures 1C and 1D; reflectance imaging setup shown in Method S1; laser invariance demonstrated in Figure S1). From these training data, ground-truth pixel-classification labels representing "background," "nuclei," and "cytoskeleton" classes are easily assembled by binary thresholding the training slide's fluorescence information (Figure 1E). These labels are then used to train a simple U-Net neural network¹³ (Method S1; Figure S2) to output the probability that pixels in the reflectance image belong to each of the classifications (demonstrated using MATLAB, Python, or standalone Windows software; Method S1). Because of pixel-wise averaging of any error in the binary representations of staining used as the around truth, the probability maps outputted by the network exhibit smooth intensity gradients that flow between classifications (Figure 1F). The nature of these images enables them to serve directly as inputs for segmentation of individual cell objects (Figures 1G and 1H; Video S1). This process can be achieved using the user-friendly CellProfiler software,¹² providing a flexible and accessible route to bioimage analysis, feature extraction, and reproducible workflow dissemination. After training, subsequent experimental samples only require the reflectance image to obtain the cell segmentation, leaving the fluorescence spectrum entirely available (Figure S3) for any form of spectral interrogation or combination of fluorescent markers (Figure 1H).

To probe the accuracy of the cell segmentation achieved while exploring the compatibility of the approach with different tissue types, we moved on to tissue sections from intestinal Peyer's patches (Figure 2A), which are key players in the orchestration of mucosal and systemic antibody responses for the microbiome, food, and oral vaccines. To do this, the cell segmentation achieved in CellProfiler using either the fluorescence information (Figure 2B) or the label-free probability maps from the reflectance data (Figure 2C) was compared against the results of careful manual annotation (Figure 2A) using the intersec-

Using a parallel tissue section to Figure 2 (i.e., with cell segmentation accuracy established), we next considered the ability of the approach to simplify image-based cell profiling. As such, we replicated a recently published experiment² that had previously required dedicated nuclei and actin fluorescent stains to achieve accurate single-cell and nearest-cell-neighbor measurements. Mouse Peyer's patch tissue sections were dual immunolabeled for CD11c (identifying mononuclear phagocytes, typically antigen-presenting cells) and for CD3 as a pan T lymphocyte marker. Alongside this, using tissue-matched serial sections, secondaryonly controls, isotype controls, and fluorescence-minus-one controls were prepared to inform on background, non-specific antibody binding and fluorescence cross-talk, respectively (see STAR Methods). In half of the previously required imaging time, each tissue section was tile scanned for fluorescence information with concomitant collection of reflected light. Figure 3A exemplifies the outcome, with a region of interest placed around the lymphoid tissue. Guided by the control data, CD11c⁺ and CD3⁺ cell populations were built using single-cell fluorescence measurements and simple, flow cytometry-type gating (Figures 3B-3E). As found previously,² a second sequential gate on the area occupied by fluorescence within each cell (Figures 3C and 3E) helped to reduce "bystander-positive" events caused by the surface-located fluorescence spanning segmented cell outlines into immediately adjacent cell objects (Figure 3F). From this simplified experiment-now using just two labels instead of the four previously required-diverse information regarding cell location. expression, and nearest-cell-neighbor relationships was obtainable (Figures 3G-3L). Interestingly, a population of highly juxtaposed, CD11c-CD3 neighboring cells that still identified positive for both markers after bystander removal were identifiable, suggesting a high likelihood of cell-cell interaction (Figure 3G). Visually intuitive cell expression maps for CD11c and CD3 could also be assembled (Figures 3H and 3I). Use of the label-free approach also enabled identification and segmentation of all of the unlabeled (i.e., CD11c⁻/CD3⁻) cells, which, in the Peyer's patch environment, predominantly represent B lymphocytes.¹⁴ Hence, the spatial distribution of antigen-presenting cell (APC)-T, APC-B, and T-B lymphocytes that were within interactive distances of one another as nearest-cell neighbors could also be isolated and mapped from the label-free objects (Figure 3J-3L). In this regard, comparing Figures 3J and 3K, a predominance of APC-B interactions, as opposed to APC-T interactions, were observed within the immunoactive subepithelial dome region of the tissue.¹⁵

In addition to frozen samples, tissue specimens are also commonly archived in formalin-fixed paraffin-embedded (FFPE) format. As a final 2D experiment, we therefore considered if the approach was transferable to this section type. Of note, F-actin staining using phalloidin conjugates is known to fail in FFPE sections because the actin cytoskeleton is degraded by solvent exposures incurred during FFPE processing.² The cell outline ground-truth fluorescence labeling on the training slide was therefore switched to cell membrane (i.e., phospholipid) staining using

Report





Figure 1. Label-free cell segmentation of tissue microscopy image data collected by routine confocal microscopy

(A–D) Image data (here, mouse splenic tissue) for initial network training are obtained from serial tissue sections stained for (A) nuclei (Hoechst 33342) and (B) cytoskeletal f-actin (phalloidin-AlexaFluor 647) while simultaneously collecting (C and D) reflected laser excitation light by detector placement close (± 5 nm) to the excitation wavelength.

(E) Binary pixel-classification labels representing "background," "nuclei," and "cytoskeleton" classes are created by thresholding the fluorescence data.

(F) A neural network using a simple U-Net architecture is trained to output the probability that pixels in the reflectance image belong to each of these classes. (A)–(F) show zoomed insets of the exact same image region. Comparing across these insets, the outputted probability maps (F) exhibit consistent intensities across each image field, with clear gradients that flow between the individual classifications. This enables easy, consistent instance segmentations of individual clear classes.

(G and H) For subsequent slides, nuclei and actin stains are no longer required as the cell segmentation is achieved direct from the reflectance information via the probability map images. This establishes the cell segmentation while leaving the entire detection spectrum free for fluorescence-based analyses. For example, (H) shows the approach operating with CD3-eFluor450, CD4-PE, and CD11c-eFluor660 immunofluorescence conjugates utilizing the spectral bandwidth previously occupied by the nuclei (Hoechst 33342) and actin (phalloidin-AlexaFluor 647) stains. The label-free cell segmentation is overlaid.

(H) Insets demonstrate successful label-free cell segmentation of both CD marker-stained and entirely unstained cells in both red (green inset) and white (gray inset) pulp tissue regions.

(A–H) Main image scale bars: 250 $\mu m,$ and inset image scale bars: 10 $\mu m.$



Report



Figure 2. Assessing cell segmentation accuracies using mouse Peyer's patch tissue

(A) Hand-drawn cell segmentation performed using nuclei/actin fluorescence information for the tissue region indicated by the yellow box in the wider, tilescanned image.

(B and C) Automated cell instance segmentations for the same image-region as (A) using either (B) the raw nuclei and actin fluorescence data or (C) the label-free probability maps obtained from the neural network using reflectance alone as input (image data from this tissue section were unseen during training).

(D and E) Cell-object intersection-over-union (IOU) score distributions comparing the hand-drawn cell segmentations shown in (A) against the automated cell segmentations shown in (B and C) using either (D) fluorescence or (E) label-free information.

(F) Example hand-drawn versus label-free cell segmentation comparisons and IOU scores.

The positions of each cell in the source images are shown by the cell-object numberings in (A), (C), and (F). An IOU score of 1 represents perfect per-pixel overlap between hand-drawn and automated cell segmentations. (F) Within the comparison presented here, scores \geq 0.6 are seen to represent a good match, approaching the limit of hand-drawing accuracy given the relatively low resolution of the source image data. (A–C) Scale bars, 100 µm. (F) Scale bar, 10 µm.

fluorescently conjugated wheat germ agglutinin (WGA). Despite a notably different reflectance signal (presumably due to cytoskeletal degradation), a segmentable relationship between the reflectance information and the WGA staining was learnable (Figure S4). Encouragingly, similar IOU scores (FFPE median score = 0.74) were attainable to those achieved using frozen sections (0.72) (Figures S4 and 2). In this way, segmentations based on training exemplifications from a different fluorescence label unlocked this important section type to the label-free technique.

With the capability of our approach established in 2D, we moved forward to 3D imaging with the goal of retrieving entirely label-free segmentations for all cells in imaged volumes (Figure 4). Previously, T cell clustering in secondary lymphoid tissues (lymph nodes) and the role that FOXP3⁺ regulatory T cells play in suppressing potentially autoreactive T cells were demonstrated using 3D imaging and the "histocytometry" approach.⁴ In that work, segmentations were achieved for cells expressing a fluorescent marker—but not for unlabeled cells. Here, a simple extension to a 3-D U-Net architecture (Method S1) enabled the generation of probability maps from z stacked reflectance information that were easily segmentable into 3D cell objects using CellProfiler 4 (Figures 4A–4H). As before, cell segmentation ac-

curacies in the xy, zy, and xz dimensions were assessed against manual annotations using the IOU approach (Figure S5). Encouragingly, use of a 3D network leveraging data across multiple z planes simultaneously improved the segmentation accuracies (median IOU scores xy = 0.84, zy = 0.74, xz = 0.78) achievable relative to the 2D network results (xy = 0.72) (Figures S5 and 2).

To test the ability of the approach to retrieve single-cell and nearest-neighbor relationships in 3D, immunofluorescence data for FOXP3 or isotype control were collected (Figure 4I). As in 2D, flow cytometry-type gating established the FOXP3⁺ cell population in situ (Figures 4J-4L). FOXP3⁺ cells were, indeed, isolatable as cell populations of independent events (Figure 4M), and harnessing the segmentation information of the unlabeled cells further allowed 3D identification and visualization of all nearest-cell neighbors with interactive potential (Figure 4N). In this way, individual cells, with or without "touching" nearestcell neighbors, could be isolated and examined as independent populations in a manner similar to imaging flow cytometry,^{16,17} with the additional ability to rotate, cut away, and consider objects and their contents from any angle (Figure 4M). Moreover, with tissue-relevant localization retained, the full in vivo cell-cell environment incorporating all nearest neighbors in 3D was



Report



Figure 3. Label-free cell segmentation enables image-based cell profiling

(A) Tile-scanned mouse Peyer's patch tissue section imaged for reflectance in addition to immunofluorescence markers for CD11c (i.e., mononuclear phagocyte antigen-presenting cells) and CD3 (T lymphocytes). The yellow region of interest (ROI) represents the lymphoid tissue upon which the label-free cell segmentation approach was deployed (~16,000 cells). Outside of the ROI, the reflectance image is seen to still provide interpretable histological context.

(B–E) Flow cytometry-type gating to establish CD3⁺ and CD11c⁺ cell populations informed by secondary only, fluorescence-minus-one (fmo) and isotype singlecell fluorescence distributions obtained from label-free cell object data collected from adjacent, serial tissue sections. Due to the dense cellular packing of lymphoid tissue, (C and E) second sequential gates on the fluorescence area occupied per cell object helped to reduce (F) bystander-positive events caused by fluorescence overlap into neighboring cells.

(G) Cell map view showing the gated cell populations *in situ* using flood filling of label-free cell-objects. Juxtaposed CD11c-CD3 neighboring cells that still identified positive for both CD markers after bystander removal are shown in white.

(H and I) CD11c and CD3 expression maps with cell objects shaded into four levels (dim, low, intermediate, high) according to each segmented cell's level of immunofluorescence.

(J–L) Nearest-cell-neighbor maps simplifying the view shown in (A) to only show touching groups of cell objects according the combinations (J) CD11c⁺-CD3⁺ (i.e., APC-T), (K) CD11c⁺-CD3⁻/CD11c⁻ (i.e., APC-B), and (L) CD3⁺-CD3⁻/CD11c⁻ (i.e., T-B). In this way, the views give a sense of key cell types within interactive distances of one another. The dashed line in (J) and (K) indicates the subepithelial dome tissue region. Scale bars: 500 µm.

revealed (Figures 4L–4N). The approach was tested with tissue sections up to \sim 100 μ m in thickness. At this depth, the fluorescence from typical blue nuclear labels (e.g., Hoechst 33342 or DAPI) is attenuated by the tissue thickness and cannot serve as an input for accurate segmentation, whereas the label-free strategy still operated effectively (Figure S6).

DISCUSSION

There is a growing need for accessible means to obtain *in situ*, single-cell information from tissue images across the bioclinical sciences. A major barrier is the relatively few channels, for separate biomarkers, that conventional microscopy allows for



Cell Reports Methods Report



LF cell segm / Gated FOXP3⁺ cells / FOXP3 nearest-cell neighbours

Figure 4. 3D label-free cell segmentation of tissue microscopy image data collected by routine confocal microscopy (A-H) Stepwise exemplification of the 3D strategy using z stack image data of mouse mesenteric lymph node tissue. Outcomes at each step are displayed by (A, C, and G) orthoslice and (B, D, and H) 3D volumetric projection views, with the latter cut away to better display outcomes along the z dimension. (A and B) 3D reflectance signal.

Report

experimentation (often ≤ 6 in practice). This becomes compounded when two of these channels are required to achieve accurate cell segmentation. Imaging mass spectrometry systems, which are capable of resolving many metal-conjugated antibodies, may partially obviate these issues, but instrumentation is not widely available. In practice, to move bioclinical research beyond "representative image" reporting, 2D and 3D cell-based quantitation of tissues with standard confocal microscopy equipment must become routine, and the workflow from image data to cell features must be disseminable.^{2,4,5}

Previously, fluorescence image restoration¹⁸ and virtual in vitro cell labeling^{7,8} have been demonstrated as powerful applications of fluorescence image reconstruction by deep learning. In their seminal paper, using a custom-built multimodal reflectance microscope, Cheng et al. also showed that fluorescent stain predictions from reflected light information could be used to achieve 2D segmentation of monolayer cells in culture. Notably, however, ex vivo tissue microscopy offers a very different challenge to cell microscopy. Tissue is made up of multiple different cell types and extracellular features in dense, layerupon-layer arrangements in a way that is not present in vitro. The image information is also fundamentally different due to the histological preparation steps (fixation, embedding, and sectioning) that are different or not required for cultured cells. Here, we recognized that during confocal imaging with entirely standard equipment, there is always freely available "byproduct" reflected light. Our work now shows that this carries sufficient information, with sufficient penetration, to establish accurate 2D and 3D segmentations of cells in tissues. To do this, we use classification and probability mapping-as opposed to regressionbased fluorescent stain predictions-as intensity uniformity across the outputted probability maps is advantageous to the cell segmentation task. Moreover, we provide the software, data, and video tutorials necessary to remove the programming barrier to access, making this accessible for everyone. Because generalizability is essential to the practical utility of any method, we carefully demonstrate our approach using data from two different confocal microscopes across 40× and 63× objective lenses in three different tissues using reflectance from three different laser lines across four image resolutions (from 3.5 to 8.3 pixels per μ m). We also demonstrate the compatibility of the approach with both frozen and FFPE tissue section types.

An important aspect of the presented method is that it allows the assembly of large amounts of sample-matched training data without the need of cell annotation. This enables conventional U-Net models to be trained with bespoke exemplifications of



the task,^{13,18} maximizing performance while providing data at a scale sufficient to avoid memorization and enable rigorous cross-validation testing.¹⁹ These training data are prepared using antibody-independent affinity staining, enabling easy transfer across species² while minimizing pixel labeling errors through avoidance of non-specific binding.⁷

In this way, the presented work enables a move beyond disaggregated flow cytometry measurements and qualitative microscopy reporting by harnessing label-free information from every cell in a tissue section such that cell content and *in situ* location can be reported together. Detailed cell-cell interactions (nearestneighbor-type relationships) in complex tissue environments are achievable and will provide the bridge between deep immunological knowledge of single cell types and macroscopic tissue function. All data, code, and methodological steps are available for download alongside detailed video tutorials demonstrating deployment in Python, MATLAB, or provided standalone software for Windows.

Limitations of the study

This technique relies on the existence of a relationship between the reflectance signal and the fluorescence information used to determine the ground truth.^{18,20} Moreover, this relationship must describe the cellular structure in a manner that enables accurate cell-object segmentations. Here, we demonstrate that label-free predictions of cytoskeletal or cell membrane structure enable this from frozen and paraffin-embedded tissue sections for diverse lymphoid tissues (spleen, Peyer's patch, and mesenteric lymph node), where cell relationships are so important in establishing fundamental biology including responses to infection, vaccination, and carcinogenesis. However, this approach may not work in every tissue type as it is dependent upon the specific structural morphology of cell and tissue and the resultant optical scattering coefficients.

STAR***METHODS**

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
 - Lead contact
 - Materials availability
 - Data and code availability
- EXPERIMENTAL MODEL AND SUBJECT DETAILS

⁽C–F) Label-free probability maps outputted from a 3D U-Net neural network (C and D) with insets demonstrating the (E) label-free probability map representation for the cytoskeleton compared with (F) fluorescent cytoskeletal F-actin staining.

⁽G and H) 3D label-free cell segmentation results where (repeated) filled colors represent individual cell objects.

⁽I–N) Validation of the 3D label-free approach using tissue sections immunolabelled for FOXP3 or matched isotype control with no nuclei or actin staining present. (J and K) Flow cytometry-type gating using cell-object fluorescence distributions to establish FOXP3⁺ events from cell intensity and fluorescence volume information.

 ⁽L) 3D projection of the label-free cell segmentation results. On the left, gated FOXP3⁺ cells are identified using red surface overlays. On the right, both FOXP3⁺ events (red) and their touching nearest-cell neighbors (cyan) are shown *in situ*. (M), 3D projections of individual FOXP3⁺ cell objects cut out and montaged from the label-free segmentation. An intranuclear core of FOXP3 staining is visible surrounded by the label-free probability map for the cytoskeleton classification.
 (N) 3D projections of individual FOXP3⁺ cell objects and their touching nearest-cell neighbors cut out and montaged using the label-free cell segmentation.
 (A, C, and G) Scale bars: 50 μm.

- Murine tissues
- METHOD DETAILS

CelPress

- Tissue sectioning
- Immunofluorescence labeling
- Antibody controls
- Confocal microscopy
- QUANTIFICATION AND STATISTICAL ANALYSIS
 - Label-free cell segmentation workflow
 - O 2D U-NET
 - O 3D U-NET
 - O 2D/3D U-NET: Standalone windows software
 - Segmentation accuracy
 - Single-cell data extraction
 - Image analysis and data visulisation
- ADDITIONAL RESOURCES

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j. crmeth.2023.100398.

ACKNOWLEDGMENTS

The authors acknowledge the UK Engineering and Physical Sciences Research Council (grant EP/N013506/1), the UK Biotechnology and Biological Sciences Research Council (grant number BB/P026818/1), and the UK Medical Research Council (grant number MR/R005699/1) for supporting the work. This research was also funded in part by the Wellcome Trust (grant number 108045/Z/15/Z). For the purpose of open access, the author has applied a CC BY public copyright license to any author accepted manuscript version arising from this submission. J.W.W. is grateful to Girton College and the University of Cambridge Herchel-Smith Fund for supporting him with fellowships.

AUTHOR CONTRIBUTIONS

J.W.W., J.J.P., and P.R. conceived the concept. J.W.W., J.J.P., R.E.H., C.E.B., and P.T. designed the biological experiments. J.W.W., J.R., R.E.H., and M.M. carried out the immunofluorescence labeling and collected the image data. J.W.W., P.R., C.M.C.G., and H.D.S. designed and optimized the deep-learning method. J.W.W. and P.R. wrote the MATLAB implementation. C.M.B. and J.W.W. wrote the Python implementation. P.R. and J.W.W. wrote the standalone software. J.W.W., J.P., H.D.S., and P.R. wrote the manuscript in close collaboration with all authors. All authors reviewed, contributed in full, and approved the final version of the manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

INCLUSION AND DIVERSITY

We support inclusive, diverse, and equitable conduct of research.

Received: May 18, 2022 Revised: October 14, 2022 Accepted: January 11, 2023 Published: February 2, 2023

REFERENCES

 Da Silva, C., Wagner, C., Bonnardel, J., Gorvel, J.P., and Lelouard, H. (2017). The peyer's patch mononuclear phagocyte system at steady state and during infection. Front. Immunol. 8, 1254. https://doi.org/10.3389/ fimmu.2017.01254. Report

Cell Reports Methods

- Wills, J.W., Robertson, J., Summers, H.D., Miniter, M., Barnes, C., Hewitt, R.E., Keita, Å.V., Söderholm, J.D., Rees, P., and Powell, J.J. (2020). Image-based cell profiling enables quantitative tissue microscopy in gastroenterology. Cytometry A. 97, 1222–1237. https://doi.org/10.1002/cyto.a. 24042.
- Stoltzfus, C.R., Filipek, J., Gern, B.H., Olin, B.E., Leal, J.M., Wu, Y., Lyons-Cohen, M.R., Huang, J.Y., Paz-Stoltzfus, C.L., Plumlee, C.R., et al. (2020). CytoMAP: a spatial analysis toolbox reveals features of myeloid cell organization in lymphoid tissues. Cell Rep. *31*, 107523. https://doi.org/10. 1016/j.celrep.2020.107523.
- Liu, Z., Gerner, M.Y., Van Panhuys, N., Levine, A.G., Rudensky, A.Y., and Germain, R.N. (2015). Immune homeostasis enforced by co-localized effector and regulatory T cells. Nature 528, 225–230. https://doi.org/10. 1038/nature16169.
- Gerner, M.Y., Kastenmuller, W., Ifrim, I., Kabat, J., and Germain, R.N. (2012). Histo-cytometry: a method for highly multiplex quantitative tissue imaging analysis applied to dendritic cell subset microanatomy in lymph nodes. Immunity 37, 364–376. https://doi.org/10.1073/pnas.1708981114.
- Robertson, D., Savage, K., Reis-Filho, J.S., and Isacke, C.M. (2008). Multiple immunofluorescence labelling of formalin-fixed paraffin-embedded (FFPE) tissue. BMC Cell Biol. 9, 13. https://doi.org/10.1186/1471-2121-9-13.
- Christiansen, E.M., Yang, S.J., Ando, D.M., Javaherian, A., Skibinski, G., Lipnick, S., Mount, E., O'Neil, A., Shah, K., Lee, A.K., et al. (2018). In silico labeling: predicting fluorescent labels in unlabeled images. Cell *173*, 792– 803.e19. https://doi.org/10.1016/j.cell.2018.03.040.
- Cheng, S., Fu, S., Kim, Y.M., Song, W., Li, Y., Xue, Y., Yi, J., and Tian, L. (2021). Single-cell cytometry via multiplexed fluorescence prediction by label-free reflectance microscopy. Sci. Adv. 7, eabe0431. https://doi. org/10.1126/sciadv.abe0431.
- Vicar, T., Balvan, J., Jaros, J., Jug, F., Kolar, R., Masarik, M., and Gumulec, J. (2019). Cell segmentation methods for label-free contrast microscopy: review and comprehensive comparison. BMC Bioinf. 20, 360. https://doi.org/10.1186/s12859-019-2880-8.
- Cameron, W.D., Bennett, A.M., Bui, C.V., Chang, H.H., and Rocheleau, J.V. (2021). Leveraging multimodal microscopy to optimize deep learning models for cell segmentation. APL Bioeng. 5, 016101. https://doi.org/10. 1063/5.0027993.
- Lee, M., Lee, Y.H., Song, J., Kim, G., Jo, Y., Min, H., Kim, C.H., and Park, Y. (2020). Deep-learning-based three-dimensional label-free tracking and analysis of immunological synapses of CAR-T cells. Elife 9, e49023. https://doi.org/10.7554/eLife.49023.
- Carpenter, A.E., Jones, T.R., Lamprecht, M.R., Clarke, C., Kang, I.H., Friman, O., Guertin, D.A., Chang, J.H., Lindquist, R.A., Moffat, J., et al. (2006). CellProfiler: image analysis software for identifying and quantifying cell phenotypes. Genome Biol. 7, R100. https://doi.org/10.1186/gb-2006-7-10-r100.
- Falk, T., Mai, D., Bensch, R., Çiçek, Ö., Abdulkadir, A., Marrakchi, Y., Böhm, A., Deubner, J., Jäckel, Z., Seiwald, K., et al. (2019). U-Net: deep learning for cell counting, detection, and morphometry. Nat. Methods 16, 67–70. https://doi.org/10.1038/s41592-018-0261-2.
- De Jesus, M., Ahlawat, S., and Mantis, N.J. (2013). Isolating and immunostaining lymphocytes and dendritic cells from murine Peyer's patches. J. Vis. Exp., e50167. https://doi.org/10.3791/50167.
- Reboldi, A., Arnon, T.I., Rodda, L.B., Atakilit, A., Sheppard, D., and Cyster, J.G. (2016). IgA production requires B cell interaction with subepithelial dendritic cells in Peyer's patches. Science 352, aaf4822. https://doi.org/ 10.1126/science.aaf4822.
- Eulenberg, P., Köhler, N., Blasi, T., Filby, A., Carpenter, A.E., Rees, P., Theis, F.J., and Wolf, F.A. (2017). Reconstructing cell cycle and disease progression using deep learning. Nat. Commun. *8*, 463. https://doi.org/ 10.1038/s41467-017-00623-3.

Cell Reports Methods Report



- Blasi, T., Hennig, H., Summers, H.D., Theis, F.J., Cerveira, J., Patterson, J.O., Davies, D., Filby, A., Carpenter, A.E., and Rees, P. (2016). Labelfree cell cycle analysis for high-throughput imaging flow cytometry. Nat. Commun. 7, 10256. https://doi.org/10.1038/ncomms10256.
- Weigert, M., Schmidt, U., Boothe, T., Müller, A., Dibrov, A., Jain, A., Wilhelm, B., Schmidt, D., Broaddus, C., Culley, S., et al. (2018). Contentaware image restoration: pushing the limits of fluorescence microscopy. Nat. Methods 15, 1090–1097. https://doi.org/10.1038/s41592-018-0216-7.
- Moen, E., Bannon, D., Kudo, T., Graf, W., Covert, M., and Van Valen, D. (2019). Deep learning for cellular image analysis. Nat. Methods 16, 1233–1246. https://doi.org/10.1038/s41592-019-0403-1.
- Ounkomol, C., Seshamani, S., Maleckar, M.M., Collman, F., and Johnson, G.R. (2018). Label-free prediction of three-dimensional fluorescence images from transmitted-light microscopy. Nat. Methods 15, 917–920. https://doi.org/10.1038/s41592-018-0111-2.
- Caicedo, J.C., Cooper, S., Heigwer, F., Warchal, S., Qiu, P., Molnar, C., Vasilevich, A.S., Barry, J.D., Bansal, H.S., Kraus, O., et al. (2017). Dataanalysis strategies for image-based cell profiling. Nat. Methods 14, 849–863. https://doi.org/10.1038/nmeth.4397.
- Peng, H., Bria, A., Zhou, Z., lannello, G., and Long, F. (2014). Extensible visualization and analysis for multidimensional images using Vaa3D. Nat. Protoc. 9, 193–208. https://doi.org/10.1038/nprot.2014.011.



STAR*METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Antibodies		
Rat anti-mouse CD3-EF450	Thermo Fisher	Cat #48-0032-82
Rat anti-mouse CD4-PE	Thermo Fisher	Cat #12-0041-83
Hamster anti-mouse CD11c-EF660	Thermo Fisher	Cat #50-0114-82
Rat anti-mouse FOXP3-EF660	Thermo Fisher	Cat #50-5773-82
Rabbit anti-mouse CD3	Abcam	Cat #AB5690
Hamster anti-mouse CD11c	Abcam	Cat #AB33483
Goat anti-Rabbit IgG (H + L) Alexa Fluor 568	Thermo Fisher	Cat #A-11011
Goat anti-Hamster IgG (H + L) Alexa Fluor 488	Thermo Fisher	Cat #A-11008
Biological samples		
Frozen C57BL/6J mouse spleen sections	This paper	N/A
Frozen C57BL/6J mouse Peyer's patch sections	This paper	N/A
Frozen C57BL/6J mouse mesenteric lymph	This paper	N/A
node sections		
Formalin-fixed paraffin embedded C57BL/6J mouse Peyer's patch sections	This paper	N/A
Chemicals, peptides, and recombinant proteins		
Hoescht 33,342	Thermo Fisher	Cat #H3570
Phalloidin-AlexaFluor 647	Thermo Fisher	Cat #A22287
Wheat germ agglutinin-AlexaFluor 555	Thermo Fisher	Cat #W32464
Deposited data		
Raw and analyzed microscopy data	This paper	https://www.ebi.ac.uk/biostudies/studies/
	· • • •	
	- F - F -	S-BSST742
Experimental models: Organisms/strains		S-BSST742
Experimental models: Organisms/strains Mouse: C57BL/6J	Charles River	S-BSST742 Cat #027
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms	Charles River	S-BSST742 Cat #027
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code	Charles River This paper	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code	Charles River This paper This paper	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software	Charles River This paper This paper This paper	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later)	Charles River This paper This paper This paper MathWorks	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3	Charles River This paper This paper This paper MathWorks MathWorks	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/help/deeplearning/
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4	Charles River Charles River This paper This paper This paper MathWorks MathWorks MathWorks	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/inege.html
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4 Computer Vision Toolbox 10.1	Charles River Charles River This paper This paper MathWorks MathWorks MathWorks MathWorks	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/computer- vision.html
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4 Computer Vision Toolbox 10.1 Python 3.6	Charles River This paper This paper This paper MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/computer- vision.html https://www.python.org/downloads/release/ python-360/
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4 Computer Vision Toolbox 10.1 Python 3.6 tensorflowgpu 1.9.0	Charles River This paper This paper This paper MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks MathWorks	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/computer- vision.html https://www.python.org/downloads/release/ python-360/ https://pypi.org/project/tensorflow-gpu/1.9.0/
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4 Computer Vision Toolbox 10.1 Python 3.6 tensorflowgpu 1.9.0 Keras 2.1.5	Charles River Charles River This paper This paper This paper MathWorks MathWorks MathWorks MathWorks python.org tensorflow.org keras.io	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/release/ python-360/ https://pypi.org/project/tensorflow-gpu/1.9.0/ https://pypi.org/project/keras/2.1.5/
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4 Computer Vision Toolbox 10.1 Python 3.6 tensorflowgpu 1.9.0 Keras 2.1.5 numpy 1.18.1	Charles River Charles River This paper This paper This paper MathWorks MathWorks MathWorks MathWorks Intersorflow.org keras.io numpy.org	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/computer- vision.html https://uk.mathworks.com/products/computer- vision.html https://www.python.org/downloads/release/ python-360/ https://pypi.org/project/tensorflow-gpu/1.9.0/ https://pypi.org/project/keras/2.1.5/ https://pypi.org/project/numpy/1.18.1/
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4 Computer Vision Toolbox 10.1 Python 3.6 tensorflowgpu 1.9.0 Keras 2.1.5 numpy 1.18.1 scipy 1.4.1	Charles River Charles River This paper This paper This paper MathWorks MathWorks MathWorks MathWorks InthWorks MathW	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/computer- vision.html https://uk.mathworks.com/products/computer- vision.html https://pypi.org/project/tensorflow-gpu/1.9.0/ https://pypi.org/project/keras/2.1.5/ https://pypi.org/project/scipy/1.4.1/
Experimental models: Organisms/strains Mouse: C57BL/6J Software and algorithms MATLAB code Python code Standalone label free prediction software MATLAB R2021a (or later) Deep Learning Toolbox 14.3 Image Processing Toolbox 11.4 Computer Vision Toolbox 10.1 Python 3.6 tensorflowgpu 1.9.0 Keras 2.1.5 numpy 1.18.1 scipy 1.4.1 pandas 1.0.3	Charles River Charles River This paper This paper This paper MathWorks MathWorks MathWorks MathWorks InthWorks MathW	S-BSST742 Cat #027 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 Method S1 or https://www.ebi.ac.uk/biostudies/ studies/S-BSST742 https://www.ebi.ac.uk/biostudies/studies/ S-BSST742 https://uk.mathworks.com/products/new_ products/release2021a.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/image.html https://uk.mathworks.com/products/computer- vision.html https://www.python.org/downloads/release/ python-360/ https://pypi.org/project/tensorflow-gpu/1.9.0/ https://pypi.org/project/keras/2.1.5/ https://pypi.org/project/scipy/1.4.1/ https://pypi.org/project/scipy/1.4.1/

(Continued on next page)

Report



SOURCE	IDENTIFIER
scikit-learn.org	https://pypi.org/project/scikit-learn/0.21.1/
scikit-image.org	https://pypi.org/project/scikit-image/0.16.2/
pillow.readthedocs.io	https://pypi.org/project/Pillow/8.4.0/
ipython.org	https://pypi.org/project/ipython/7.13.0/
opencv.org	https://pypi.org/project/opencv-python/ 4.2.0.34/
pythonhosted.org	https://pypi.org/project/javabridge/
openmicroscopy.org	https://pypi.org/project/python-bioformats/1.5.2/
matplotlib.org	https://pypi.org/project/matplotlib/3.3.4/
h5py.org	https://pypi.org/project/h5py/2.10.0/
imageio.readthedocs.io	https://pypi.org/project/imageio/2.11.0/
oracle.com	https://www.oracle.com/uk/java/technologies/ javase/jdk11-archive-downloads.html
developer.nvidia.com	https://developer.nvidia.com/ cuda-90-download-archive
developer.nvidia.com	https://developer.nvidia.com/cudnn
cellprofiler.org	https://cellprofiler.org/releases
alleninstitute.org	https://github.com/Vaa3D/release/releases/
Zeiss	https://www.zeiss.com/microscopy/en/products/ light-microscopes/confocal-microscopes.html
Leica Microsystems	https://www.leica-microsystems.com/products/ confocal-microscopes/
	SOURCE scikit-learn.org scikit-image.org pillow.readthedocs.io ipython.org opencv.org pythonhosted.org openmicroscopy.org matplotlib.org h5py.org imageio.readthedocs.io oracle.com developer.nvidia.com cellprofiler.org alleninstitute.org Zeiss Leica Microsystems

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, John W. Wills (jw2020@cam.ac.uk).

Materials availability

This study did not generate new unique reagents.

Data and code availability

- Microscopy data are publically available as of the date of publication from the BioStudies database (https://www.ebi.ac.uk/ biostudies/) under accession number S-BSST742.
- All original code (in MATLAB and Python languages) is available in the Method S1 file. All code as well as the precompiled, Windows software has also been deposited at the BioStudies database (https://www.ebi.ac.uk/biostudies/) under accession number S-BSST742 alongside screencast tutorial videos demonstrating deployment. These files are publicly available as of the date of publication.
- Any additional information required to reanalyse the data reported in this paper is available from the lead contact upon request.

EXPERIMENTAL MODEL AND SUBJECT DETAILS

Murine tissues

Spleen, ileum (containing Peyer's patches) and mesenteric lymph node tissues were collected from healthy, male C57BL/6 mice (n = 4) (8-12 week-old) sacrificed by carbon dioxide asphyxiation. Tissues for cryosection analysis were snap-frozen in isopentane cooled on dry ice before storage in liquid nitrogen until use. Tissues for FFPE processing were fixed in neutral buffered formalin (4 h) prior to transfer to tissue cassettes. Samples were embedded in paraffin by dehydrating through an aqueous ethanol series (5 min each 20%, 50% 70% (100% x2) v/v) followed by three changes of 100% xylene (30°C) then three changes of paraffin wax (62°C). All animal work complied with the University of Cambridge Ethics Committee regulations and was performed under the Home Office Project License numbers 80/2572 and P48B8DA35.



Cell Reports Methods Report

METHOD DETAILS

Tissue sectioning

Frozen tissues were transferred into the cryostat chamber (- 15° C) and acclimatised for 30 min. Samples were trimmed to remove any excess fat, and transferred to cryomolds containing pre-chilled optimal cutting temperature compound (OCT) (#00411243, VWR). Sections were cut at 25 or 100 micron thicknesses (for 2D or 3D imaging, respectively) and collected on Super-Frost Plus adhesion treated slides (#J1800AMNT, Thermo) before resting at room temperature for 2 h prior to immunofluorescence labeling. Formalin-fixed, paraffin embedded (FFPE) sections were cut at 5 μ m thickness. FFPE sections were dewaxed by baking at 60°C for 1 h prior to changing twice through xylene. Prior to fluorescence counterstaining, FFPE sections were rehydrated using a reverse ethanol series (100%, 70%, 50%, 10%; 5 min each) followed by immersion in water (1 min).

Immunofluorescence labeling

Tissue sections were ringed with hydrophobic barrier pen (Vector, #H-4000). Frozen sections were fixed using fresh 4% paraformaldehyde in 0.1 M PBS (pH 7.4) at room temperature for 10 or 20 min (25 μ m or 100 μ m sections, respectively). All subsequent steps were carried out under gentle agitation on a rotating shaker. To facilitate antibody penetration, the frozen sections were permeabilised for 2 or 4 h using 0.3% (v/v) Triton X-100 in 0.1 M PBS (pH 7.4) (25 μ m or 100 μ m sections, respectively). Sections were then blocked using 25 mM TBS (pH 7.4) supplemented with 10% (v/v) goat serum (ThermoFisher, #16210064), 2% (w/v) BSA (BioSera, #PM-T1726) and 20 mM glycine for 2 h. Primary antibodies or isotype controls were prepared in block buffer and added at 150 μ L per section for 18–24 h at 4°C. All subsequent steps took place at room temperature. Sections were washed (3 × 3 min, TBS) prior to incubation with secondary antibodies (when needed) diluted in block buffer for 4 or 8 h (see Table S1 for antibody concentrations, fluorophores conjugations and manufacturer information) (25 μ m or 100 μ m sections, respectively). After washing (3 × 3 min, TBS) frozen sections destined for the provision of training data underwent nuclear and f-actin counterstaining using 2 μ g/mL Hoechst 33,342 (#H3570, Thermo) and 500 nM phalloidin-AlexaFluor 647 (#A22287, Thermo) in TBS for 1 h. The cellular structure of the FFPE sections was counterstained by labeling cell membranes using 20 μ g/mL wheat-germ agglutinin (WGA) conjugated with Alexa Fluor 555 (#W32464, Thermo). After counterstaining, all sections were washed for a final time (1 × 3 min, TBS) before mounting with #1.5 coverslips in Prolong Glass mountant (#P36980, Thermo).

Antibody controls

Three antibody controls commonly used by the flow cytometry community were measured in tissue-matched serial sections. Secondary-only controls received just the secondary antibody in absence of any primary antibody. Any signal in the collection channel for this control thus represented endogenous tissue autofluorescence or contributions from the fluorophore-conjugated secondary antibody binding non-specifically in the tissue section. Fluorescence-minus-one (FMO) controls contained all of the fluorescent stains – bar the one under quantification. Here, any signal in the collection channel was typically from 'spill over' from the other fluorophores into this empty channel. Finally, 'isotype controls' switched out the primary antibody for an irrelevant antibody (*i.e.*, raised against an antigen not present in the sample) but otherwise identical (*i.e.* same isoclass) to the primary antibody. Here, non-specific binding of this irrelevant primary antibody or capture by, *e.g.*, Fc receptors led to signal in the collection channel, informing on the level of nonspecific binding.

Confocal microscopy

The label-free strategy was developed using image data collected from two commonplace (Leica SP8/Zeiss LSM 780) laser scanning confocal microscopy platforms. The SP8 was inverted configuration, whilst LSM780 was upright. No modifications from standard were necessary to enable the presented approach. Detailed instructions for setting up reflected light collection are provided in Method S1. Image data for the presented 2D analyses were collected using the SP8 via 40X/1.3 or 63X/1.4 oil immersion objectives. Reflectance was collected from the 488 nm laser via detector placement +/-3 nm either side of the excitation line (*i.e.*, detection in the range 485-491 nm). 3D data were collected using the LSM 780 via the 40X/1.3 oil immersion objective. Reflectance was collected placement +/-9 nm either side of the excitation line (*i.e.*, 552-600 nm). Tilescans were conducted with 10% edge overlap to facilitate registration. Details of the tissue specimen, image dimensions and pixel/voxel densities are provided for all image data in Table S1.

QUANTIFICATION AND STATISTICAL ANALYSIS

Label-free cell segmentation workflow

Neural network training data was collected from parallel tissue sections to those undergoing immunofluorescence labeling. Training data was achieved by collecting fluorescence images for nuclei and cytoskeletal f-actin (frozen sections) or nuclei and the cell membrane (FFPE sections) (fluorescence staining described above) alongside the 'paired' reflectance signal. Binary pixel classification labels were assembled by thresholding the fluorescence information to create pixel label classes representing 'nuclei', 'cytoskeleton' and 'background' classifications. Because of the paired nature of the training and test image-data (*i.e.,* collected using the same microscope settings) input reflectance data were rescaled in the zero-one interval with no contrast

Report



adjustment prior to inputting into the 2D or 3D U-Nets (architectures shown, Method S1). Using cross-entropy loss, the networks were then trained to output the probability that pixels in the reflectance image belonged to each of the classifications with the probability image from the loss function serving as the direct input for cell segmentation. Network training, optimisation and validation testing was conducted using MATLAB R2021a and the Deep Learning, Image Processing and Computer Vision toolboxes (described, Method S1). Scripts for running the 2D and 3D U-Nets were also written for Python 3 using keras/TensorFlow-gpu 1.9 (TensorFlow install guide and U-Net scripts described, Method S1). The probability map images outputted by the U-Net networks were segmented into 2D or 3D cell objects using marker-controlled watershed algorithms deployed in CellProfiler¹² (version 4.1.3) (instructions for installing CellProfiler and running the 2D and 3D image analysis pipelines are provided in Method S1). In all instances, neural networks were trained on data from one tissue section before validation testing using data collected from an entirely different tissue section.

2D U-NET

Reflectance data were passed to the network as patches with dimensions $256 \times 256 \times 1$ (x, y, channels) with augmentation by x/y reflection and rotation. The three-class U-Net architecture used an encoder depth of 4 with 64 filters in the first layer (shown, Method S1). Complete up-convolutional expansion was used to provide probability maps of the same size as the input images. Training lasted for 50 epochs (frozen sections) or 150 epochs (FFPE sections) using a batch size of 12 with zero-center normalisation (demonstrated, Video S1). Training was optimised using stochastic gradient descent using cross-entropy loss. The initial learning rate was 0.05, dropping every 10 epochs by 0.1 under momentum 0.9 and L2 regularisation 1×10^{-4} . Patches were shuffled every epoch.

3D U-NET

Reflectance data were patched through the network with input dimensions $64 \times 64 \times 64 \times 14$ (x, y, z, channels) and augmentation by x/y reflection and rotation. The three-class U-NET architecture used an encoder depth of 4 with 64 filters in the first layer (shown, Method S1). Complete up-convolutional expansion was used to output probability maps of the same dimensions as the source microscopy data. Training lasted for 150 epochs using a batch size of 8 with zero-center normalisation. Training was optimised under ADAM using cross-entropy loss. The initial learning rate was 5×10^{-4} , dropping every 5 epochs by 0.95 under L2 regularisation 1×10^{-4} . Patches were shuffled every epoch.

2D/3D U-NET: Standalone windows software

The label-free prediction software (described, Method S1) for Windows was built in MATLAB R2021a using the MATLAB App Designer and MATLAB Compiler. This enables 2D/3-D U-NET training and deployment via a simple graphical user interface removing the need for programming expertise.

Segmentation accuracy

Using the Jaccard index (intersection over union) approach, 2D and 3D label-free cell segmentation accuracies were assessed by comparing pixel positions within automatically segmented cell objects against those inside manually-drawn cell outline annotations. To assess the 3D segmentations, annotations of XY as well as ZY and XY dimensions were used to fully explore the validity of the segmented cell objects along all three dimensions. The Jaccard index was calculated as:

$$J(A,M) = \frac{|A \cap M|}{|A \cup M|} = \frac{|A \cap M|}{|A| + |M| - |A \cap M|}$$
(Equation 1)

Where, J is the Jaccard distance for two sets containing pixel positions for the automated segmentation (A) and the manual annotation (M) respectively. A score of 0 represents no overlap (*i.e.*, false negative) whereas 1 represents exact pixel-for-pixel overlap. It is acknowledged that this approach is a relatively harsh success measure and that a score of \sim 0.7 indicates a good segmentation result.¹³ This is due in-part to the inaccuracies that are inevitably present even in the human annotated data (*e.g.*, due to outline smoothing, ambiguity in determining the precise position of each cell's boundary from the fluorescent staining information and available image resolution *etc.*).

Single-cell data extraction

After cell segmentation, subsequent CellProfiler modules enabled image preprocessing and cell feature extraction. 2D and 3D CellProfiler workflows are demonstrated in the Method S1. Immunofluorescence channels were thresholded at the level required to remove ~95% of fluorescence in tissue-matched, secondary antibody-only control images.² Fluorescence intensity values per cell, alongside cell size and shape features were then measured for all channels. Integration of binarized immunofluorescence images was used to measure the fluorescence area/volume.

Image analysis and data visulisation

Following recommended best practice,²¹ cell-objects lying outside of the fifth or 95th percentiles by area (2D analyses) or volume (3D analyses) were discarded. Nearest-cell neighbors to gated startpoint cells were identified using a spherical structuring element to





dilate the startpoint cell's boundary by 3 pixels before identifying neighboring objects subsequently eroded. Immunofluorescence visualisations and gated cell-object surface overlays were created using the freely available Vaa3D software.²²

ADDITIONAL RESOURCES

Image-data, code and screencast tutorials demonstrating deployment of the described label-free cell segmentation method using MATLAB (R2021a, using Deep Learning, Image Processing and Computer Vision toolboxes), Python 3 (using keras/Tensorflow-gpu 1.9) or via precompiled, standalone software for Windows are downloadable from the BioStudies database (https://www.ebi. ac.uk/biostudies/) under accession number S-BSST742.

Cell Reports Methods, Volume 3

Supplemental information

Label-free cell segmentation

of diverse lymphoid tissues in 2D and 3D

John W. Wills, Jack Robertson, Pani Tourlomousis, Clare M.C. Gillis, Claire M. Barnes, Michelle Miniter, Rachel E. Hewitt, Clare E. Bryant, Huw D. Summers, Jonathan J. Powell, and Paul Rees



Figure S1 related to Figure 1 – Comparing label-free probability maps using reflectance data obtained using 488 nm or 638 nm laser excitation. The choice of excitation wavelength for generating the reflectance signal has minimal influence on the probability maps obtained from the network (left versus middle; difference shown right). The user can therefore reasonably use whatever is available on their individual microscope. For some applications, choosing a longer excitation wavelength may reduce fluorophore photobleaching / improve tissue penetrance and reflectance recovery - for instance, during 3-D Z-stack imaging in thicker tissue specimens.



Figure S2 related to Figure 1 – Number of first-encoder filters and encoder depth optimisation to define the best performing 2-D Unet model. Bars represent normalised training time whilst circles indicate label-free cell segmentation accuracies (assessed by Jaccard index). The best performing model used an encoder depth of 4 with 64 filters at the level of the first encoder (indicated, **). Of note, using 32 filters instead of 64 can achieve a training speed up of ~50% for a negligible (~1%) decrease in segmentation accuracy (indicated, *). Increasing the encoder depth to 5 did not further improve cell segmentation accuracies (bottom row).



Figure S3 related to Figure 1 – Spectral bandwidth saving achieved by the label-free cell segmentation strategy. a, Emission spectra for Hoechst 33342 and AlexaFluor 647 as might typically be used to delineate cell nuclei and cell cytoskeletons when carrying out fluorescence-based cell segmentation. b, Harnessing reflectance information, the label-free cell segmentation method described here removes the need for these fluorescence stains leaving the spectrum entirely open for sensitive experimental measurements with single-cell quantification.



Figure S4 related to Figure 2 – Label-free cell segmentation of confocal microscopy image-data collected from formalin-fixed, paraffin embedded tissue sections. a, In frozen cryostat sections, f-actin staining using phalloidin conjugates clearly delineates cell outlines providing ground truth to enable the presented label-free cell segmentation approach. b, In contrast, in formalin-fixed paraffin embedded (FFPE) tissue sections, phalloidin staining fails because solvent exposure during the fixation and paraffin embedding process degrades the actin cytoskeleton. c, Demonstrated here using murine Peyer's patch tissue sections, successful ground truth labelling can be restored in the FFPE section-type by switching to cell membrane (i.e., phospholipid) staining using wheat germ agglutinin (WGA) fluorescence conjugates. d/e Comparison of the reflectance signal from the frozen and FFPE section-types. Cytoskeletal degradation appears to change the reflectance images observed from the FFPE tissue: the faint trace of the cell outlines visible in the frozen sections is no longer apparent and instead the intracellular regions appear to exhibit the highest reflectance signal. f-h Despite this, a relationship between the reflectance signal and a WGA-delineated ground truth is still determinable by the neural network allowing (g/h) successful label-free cell segmentation direct from the reflectance signal. h, Intersection over union (IOU) score distribution comparing a (f) hand-drawn segmentation and the (g) automated, label-free cell segmentation outcome. An IOU score of 1 represents a perfect, per-pixel overlap between the hand-drawn and automated cell segmentations. Within the comparison presented here, scores ≥ 0.6 are seen to represent a good match, approaching the limits of hand-drawing accuracy. By harnessing ground truth from other fluorescence labels, the label-free strategy can operate in both FFPE and frozen tissue-types. Given that tissue archiving in FFPE format is commonplace worldwide, this finding dramatically increases the application domain of the presented label-free cell segmentation strategy. Scale bars: a/b = 20 microns; c/d = 100 microns; e/f = 75 microns; g/h = 50 microns.



Figure S5 related to Figure 4 – Assessing 3-D label-free cell segmentation accuracies using mouse Peyer's patch tissue. a, Hand-drawn cell segmentations performed using the nuclei/actin fluorescence information for Z-planes (a) 57 in the XY dimension (d) 512 in ZY dimension and (g) 512 in the XZ dimension (unseen test image-data is 512x512x114 (X,Y,Z)). b/e/h, Automated cell segmentations for the same image-regions as (a/d/g) but achieved label-free direct from the reflectance signal. c/f/i, Cell-object intersection-over-union score distributions comparing – cell-object by cell-object – the (a/d/g) hand-drawn segmentations against the (b/e/h) automated, label-free cell segmentations. An IOU score of 1 represents perfect, per-pixel overlap between the hand-drawn and automated cell segmentations. Within the comparison presented here, scores \geq 0.6 are seen to represent a good match, approaching the limits of hand-drawing accuracy. Encouragingly, the 3-D approach outperformed the segmentation accuracies achieved in 2-D (shown, Figure 2). Scale bars equal 20 microns.



Figure S6 related to Figure 4 – Fluorescence versus label-free nuclei predictions at Z-depths of ~10 and ~90 microns. Using reflectance information from a 638 nm excitation laser, the 3-D network is able to consistently recover nuclear information long after the blue nuclear stain (Hoechst 33342) has decayed from multiple scattering effects (bottom right versus middle right). The resultant pixel intensity histograms from the probability map images are extremely stable (bottom right). This is advantageous for achieving consistent, depth-invariant 3-D cell segmentation in thick tissue specimens.

Table S1 related to	Star Methods -	Antibody	and Image	Information	Table

PRIMARY ANTIBODIES	Product no	Supplier	Dilution primary	Stock concentration	Host	Secondary (detailed below)	Detection	Figure
Anti-mouse CD3-EF450	48-0032-82	Thermo Fisher	1:25	0.2 mg/mL	Rat	N/A	eFluor 450	Figure 1
Anti-mouse CD4-PE	12-0041-83	Thermo Fisher	1:25	0.2 mg/mL	Rat	N/A	R-phycoerythrin	Figure 1
Anti-mouse CD11c-EF660	50-0114-82	Thermo Fisher	1:25	0.2 mg/mL	Armenian Hamster	N/A	eFluor 660	Figure 1
Anti-mouse FOXP3-EF660	50-5773-82	Thermo Fisher	1:25	0.2 mg/mL	Rat	N/A	eFluor 660	Figure 2
Anti-mouse CD3	AB5690	Abcam	1:150	0.2 mg/mL	Rabbit	Anti-Rabbit AF568	AlexaFluor 568	Figure 3
Anti-mouse CD11c	AB33483	Abcam	1:400	0.5 mg/mL	Armenian Hamster	Anti-Hamster A488	AlexaFluor 488	Figure 3

SECONDARY ANTIBODIES	Product no	Supplier	Stock concentration	Host	Secondary dilution	Fluorophore
Goat anti-Rabbit IgG (H+L)	A-11011	Thermo Fisher	2 mg/mL	Goat	1:400	AlexaFluor568
Goat anti-Hamster IgG (H+L)	A-11008	Thermo Fisher	2 mg/mL	Goat	1:400	AlexaFluor488

Figure ID	Section type	Tissue type (Sectioning orientation)	Objective lens Magnification / Numerical aperture (Microscope)	Pixel density Pixels per micron	Voxel size x,y,z μm	Reflectance Excitation laser (Detector placement) Wavelength, nm	Train / Test image(s) (number) x/y/z dimensions Pixel dimensions	Patch size / (patches per epoch)	Approx. Training time (single NVIDIA 1080 Ti GPU)
Figure 1	Cryostat (frozen)	Mouse spleen (transverse)	40X/1.3 (Leica SP8 inverted)	5.2842	0.1892 x 0.1892	488 (485-491)	Train: (1) 7617 x 7661 Test: (1) 5766 x 5787	256x256 (1000)	268 min (50 epochs)
Figure 4	Cryostat (frozen)	Mouse mesenteric lymph node (transverse)	40X/1.3 (Zeiss LSM780 upright)	4.8177	0.2076 x 0.2076 x 0.1500	561 (558-564)	Train: (2) 1024 x 1024 x 107 Test: (1) 1024 x 1024 x 111	64x64x64 (1000)	750 min (150 epochs)
Figure S1 Figure S6	Cryostat (frozen)	Mouse ileal Peyer's patch	40X/1.3 (Leica SP8 inverted)	3.5200	0.2840 x 0.2840 x 0.346	488 (485-491) 638 (635-641)	Train: (3) 512 x 512 x 103 Test: (1) 512 x 512 x 103	64x64x64 (500)	320 min (150 epochs)
Figure 3 Video S1	Cryostat (frozen)	Mouse ileal Peyer's patch (transverse)	63X/1.4 (Leica SP8 inverted)	8.324	0.1201 x 0.1201	488 (485-491)	Train: (1) 7607 x 11253 Test: (1) 7610 x 11247	256x256 (1500)	360 min (50 epochs)
Figure S4	FFPE	Mouse ileal Peyer's patch (transverse)	40X/1.3 (Leica SP8 inverted)	7.0463	0.1419 x 0.1419	638 (635-641)	Train: (1) 11364 x 11421 Test: (1) 7641 x 7660	256x256 (2000)	502 min (50 epochs)

Methods S1 – Related to Star Methods

Table of contents -

Pages 2-5:	Setting up reflectance imaging on Leica and Zeiss confocal microscopes
Pages 6-12:	Label-free probability map generation using standalone Windows software
Pages 13-21:	Running the label-free cell segmentation deep learning scripts using MATLAB
Pages 22-29:	MATLAB code: 2-D pipeline
Pages 30-39:	MATLAB code: 3-D pipeline
Pages 40-58:	Running the label-free cell segmentation deep learning scripts using Python
Pages 59-68:	Python code: 2-D pipeline
Pages 69-83:	Python code: 3-D pipeline
Pages 84-89:	Extracting single-cell features using CellProfiler 4
Pages 90-119:	2-D CellProfiler pipeline
Pages 120-139:	3-D CellProfiler pipeline



Setting up sequential reflectance imaging using a standard Zeiss LSM780 confocal microscope.

a, Once the fluorescence imaging sequences are set up, a new track for reflectance is added to the sequential scan. **b**, The desired excitation laser for reflectance imaging is selected (here, 561 nm). The choice of laser is not particularly important, but use of longer wavelengths may reduce fluorophore photobleaching and improve penetrance / recovery in thicker specimens (shown, **Figures S1/S6**). **c**, A photomultiplier detector (here Ch1) is turned on and the range set to approximately +/- 3 nm either side of the excitation wavelength (here 558 – 564 nm was entered, but the software rounds to display ~ 557 – 566 nm). **d**, The tick-box allowing reflected light to pass to the detector is turned on. **e**, The T80/R20 beam splitter is chosen (this indicates a transmission/reflection ratio of 80:20). **f**, A low laser excitation power (here, 1%) is entered. *N.B.*, use of a reflectance light path with high laser excitation power may damage the camera, so care should be taken here. The pinhole is set to ~ 1 airy unit, yielding an optical section of around ~ 1 micron with a high numerical aperture 40X or 63X objective. **g**, Running in 'live mode', the gain is slowly increased until the reflectance signal occupies approximately 80% of the range histogram (indicated, green box). Compressing the histogram in the range indicated yields a typical 'view' of the reflectance signal from a lymphoid tissue specimen (on display in the main image window).



Setting up sequential reflectance imaging using a standard Leica SP8 confocal microscope.

a, Once the fluorescence imaging sequence(s) are set up, a new track for reflectance is added to the sequential scan (here, 'Seq 4'). **b**, The desired excitation laser for reflectance imaging is turned on (here, 488 nm). The choice of laser is not particularly important, but use of longer wavelengths may reduce fluorophore photobleaching and improve penetrance / recovery in thicker specimens (shown, **Figures S1/S6**). A low laser excitation power (*e.g.*, 1%) is also specified at this step. *N.B.*, Use of a reflectance light path with high laser excitation powers could damage the camera, so care should be taken at this step. **c**, An appropriate beam splitter is chosen for the excitation line, or, the 'Autoselect' checkbox can be ticked to set this automatically. **d**, A photomultiplier detector (here PMT1) is turned on and the range set to approximately +/- 3 nm either side of the excitation wavelength (*i.e.*, here, 485-491 nm). **e**, Running in 'live mode', the gain is slowly increased until the reflectance signal occupies approximately 80% of the available intensity range. The main window shows a typical 'view' of the reflectance signal from a lymphoid tissue specimen under this setup.



Setting up simultaneous reflectance and fluorescence imaging using a standard Leica SP8 confocal microscope.

Once fluorescence excitation and collection are configured, (a) any remaining laser line can be used for reflectance imaging (*e.g.*, here, the 488 nm line is used). The choice of laser is not particularly important, but use of longer wavelengths may reduce fluorophore photobleaching and improve penetrance / recovery in thicker specimens (shown, **Figures S1/S6**). A low laser excitation power (*e.g.*, 1%) should also be specified here. Reflectance imaging with high laser excitation powers could damage the camera, so care should be taken at this step. **b**, An appropriate beam splitter for the excitation lasers is chosen, or the 'Autoselect' checkbox ticked to enable automatic setting. **c**, A free detector (here PMT2) is turned on and the range set to approximately +/- 3 nm either side of the excitation wavelength (*i.e.*, 485-491 nm). **d**, Running in 'live mode', the gain is slowly increased until the reflectance signal occupies approximately 80% of the available intensity range. The top-right image in the main window shows a typical 'view' of the reflectance signal from a lymphoid tissue specimen under this setup. This approach allows reflectance data to be concomitantly collected alongside fluorescence without adding the additional run-time of further sequences. *N.B.*, It is worth noting that in a similar way, a laser that is already being used for fluorescence excitation may also be used to obtain reflectance data (exemplified **page below**). For example, here, PMT2 could be moved up to collect reflectance from the 638 nm laser in the range 635-641 nm. Doing this has the advantage of reducing the photon budget for the sample. However, it also necessitates that enough excitation power is being used to obtain a good reflectance signal, and that a free detector can be moved within the necessary detection range. This is not always compatible with optimal fluorescence imaging – hence the setup shown here.



Setting up 'free' reflectance imaging alongside fluorescence collection using a standard Leica SP8 confocal microscope.

Once fluorescence excitation and collection are configured, (a) any remaining detector can be used to simultaneously collect the reflectance signal from one of the excitation lasers being used to stimulate fluorescence (*e.g.*, here, 'PMT2' is used to collect reflectance from the 638 nm laser line (b) – which is also being used to excite AlexaFluor 647). This is achieved by placing the detector approximately +/- 3 nm either side of the excitation wavelength (*i.e.*, 635-641 nm). **c**, Running in 'live mode', the gain is slowly increased until the reflectance signal occupies approximately 80% of the available intensity range. The top-right image in the main window shows a typical 'view' of the reflectance signal from a lymphoid tissue specimen under this setup. Simultaneous reflectance imaging has the advantage of reducing the photon budget for the sample, as the reflectance information is effectively recovered for 'free' by harnessing scatter from a laser that is already in use. However, this setup also necessitates that enough excitation power is available to obtain a good reflectance signal (shown bottom-left in the main image window) the AlexaFluor647 detection range was narrowed (d) to ~ 700-750 nm. Where this setup cannot be accommodated one of the other options that instead use a dedicated laser for reflectance imaging should be utilised (shown, **three above pages)**.

Windows 10: Running the label-free cell segmentation pipeline using standalone software

Running this software requires:

- Windows 10 machine (NVIDIA GPU desirable for model training)
- Label-free stain prediction standalone software (BioStudies download)
- MATLAB runtime 9.11 (installs automatically alongside software see below).

Note: This software is free to install. No MATLAB license is required to run this standalone software.

Installation Steps:

1. **Install the Label Free Stain Prediction software** by running the Windows 10 installer from the BioStudies download



2. Follow through the setup procedure to install the software:



- Check 'add shortcut to the desktop':

🚪 Installation Options		_		\times
Choose installation folder:		3	53	Y
C:\Program Files\LabelFreeStainPrediction	Browse	\sim	1)	T
	Restore Default Folder	\checkmark	\bigcirc)
Add a shortcut to the desktop		Th.	A	5
		\sim	F	λ
		1	my	T
			Ż	~~
		\mathcal{H}	3	\searrow
		4 5		~
			hand	4
< Back Next >	Cancel			

- When prompted, install MATLAB runtime (free; no license is required):

🔏 Required Software	_		×
MATLAB Runtime is required.			
Choose installation folder:	MA	TLA	∆B°
C:\Program Files\MATLAB\MATLAB Runtime Browse	R	2021 b	
MATLAB and Simulink are registered trademarks of The MathWorks, Inc. Please see mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders. WARNING: This program is protected by copyright law and international treaties. Copyright 1984-2021, The MathWorks, Inc. Protected by U.S. and other patents. See MathWorks.com/patents			
< Back Next > Cancel	📣 Ma	athWorl	CS*

- click 'install' to install the label free stain prediction software and MATLAB runtime 9.11

a Confirmation	_		×
LabelFreeStainPrediction will be installed in: C:\Program Files\LabelFreeStainPrediction LabelFreeStainPrediction requires MATLAB Runtime R2021b. MATLAB Runtime R2021b will be installed in: C:\Program Files\MATLAB\MATLAB Runtime\v911 Download Size: 2,730 MB		States and the states	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
< Back Install > Cancel			

3. Training 2-D or 3-D UNET model using the standalone software

-A screencast video is included with the BioStudies archive.

- -Launch the Label Free Stain Prediction software from the desktop icon
- -Example 2-D and 3-D image data and training labels are included in the BioStudies download.

MATLAB App	- 🗆 X	LabelFreeStainPrediction			- 0
Train a label-free segmentation model Load reflectance image Image file Image rstal Reger stal Reger stal Reger stal C Reg	Generate label-free predictions Load pretrained model Pretrained model life Pretrained model information: Number discolution 0 8000 000 0 0 0 0 0 0 0 0 0 0 0 0 0 0				
Load nuclei labels. Nuclear label file Load cell outline labels: C	Load reflectance image:	Select File to Open $\label{eq:entropy} \leftarrow \ \ \rightarrow \ \ \ \ \ \ \ \ \ \ \$	tandalone_software_biostudies >	ບ ບັ ,⊃ Search St	itandalone_software
Cell outline tabels file Load background/other tabels Background labels file Load ROL-mask (If required) ROI mask file	Image tile 0 Image y ster 0 Image y ster 0 Image z ster 0	Organize - Newfolder Orobnice - University of Cambridge University of Cambridge Eachel Hewitt - 15 Kota for Fachel Eachel Hewitt - 15 Kota for Fachel	Name LExample_image_data QWINDOWS10 INSTALL Date created: Scrample_pretrained_m_Size_995 MB Enders FXM	Date modified 04/01/2022 20:38 04/01/2022 20:37 04/01/2022 20:37	Type File folder
Number of epochs 0 e.g., 2.0 = 50, 3.0 = Batch Size 0 e.g., 2.0 = 12, 3.0 =	50 Load ROI-mask (if required) 8 ROI mask file	This PC JD Objects Osttop Documents ↓ Downloads	5,3D_CelProfiler_pipeline	04/01/2022 21:13	File folder
TRAIN NETWORK	PREDICT LABELS	Music Pictures Gives Windows (C) Data (P)	v k		
		File name:		(*.tiff)Open	Cancel

- To Train a 2-D or 3-D UNET model to predict the probability images needed to enable the label-free segmentation, load reflectance data and matching labels for 'Nuclei', 'Cell Outlines' (e.g. actin) and 'background/other' classes.

-The label images should be in .tiff format with the stains represented as foreground (i.e., white). -2-D or 3-D data will be automatically detected.



- If desired, a ROI-mask can also be loaded to mask the outputted probability maps to the desired tissue region.

- If this isn't needed – just leave this box blank.

- Set the desired batch-size and number of training epochs and click 'Train Network'.

- After a few seconds, the loaded reflectance data and labels will be indicated in "Figure 1"

- Progress can be tracked in the Command Prompt console

- MAT formatted data and the trained model will be saved to folders on the desktop once training completes:

- Training with the example data takes ~6h on an NVIDIA 1080 GTX GPU. (Previously trained models are provided).



4. Using a pretrained 2-D or 3-D UNET model to generate probability images for label-free segmentation using unseen reflectance data.

-On the right-side of the software, **load** a pretrained model.

-This can be from Step 3 (above), or, example pretrained networks are provided in 'Folder 3' of the BioStudies download. Example unseen 'test' image-data is also provided in 'Folder 1' in 2-D and 3-D.



- Load 2-D or 3-D reflectance data:

MATLAB App	- 🗆 X		
Train a label-free segmentation model Lod reflectance image Image file Image file Image file Image file Image file Load nuclei labels Cell outline labels file Load cell outline labels Background labels file Load Rol-mask (if regured) Rol mask file	Generate label-free predictions Load pretrained model MLN_Biostudies_27-May-2021-1355 mal Pretrained model information: Number of spoors State size Load reflectance image: Image size 0 Image size 0 Image size 0	Select File to Open Ciganize V New folder Organize V New folder Organize V New folder OneDrive - University of Cambridge Sachel Howit - ISX data for Rachel Rachel Howit - ISX data for Rachel Rachel Howit - ISX data for Rachel Rachel Howit - ISX data for Rachel Documents Documents Documents Videos Videos Videos	X V O P Search 30_Test V V II O SD_MUN, Test, RO Umask
Number of epochs 0 # g, 2.0 = 50; 3.0 = 150. Batch size 0 # g, 2.0 = 12; 3.0 = 8	Load ROI-mask (if required). ROI mask file		(*.tiff) ~
TRAIN NETWORK	PREDICT LABELS		

- If desired, load a ROI-mask to limit the probability maps to the desired tissue region:
 (Leave blank if not required).

MATLAB App	– 🗆 ×		
Train a label-free segmentation model Used reflectance image Image Tile Image Tile Image 2 site Cell outline label file Load cell outline labels Cell outline labels file Cell outline labels C	Generate label-free predictions Load pretrained model MLM_Biotudies_27-May-2021-1355.mml Pretrained model information: Pretrained model information: Tretrained model information: CLEAR Load reflectance imple: Image 10: Image 10: Image 20: Image 20: Image 20: Image 20: Image 20: Image 20: Image 20:	Select File to Open Corganise Vew folder Organise Vew folder Organise Vew folder Organise Vew folder OneDrive - University of Cambridge Verviewsity of Cambridge Sachel Hewitt - NYMESIS project Sachel Hewitt - NYMESIS project Social Excel H	X V Č /P Seech 30, Test R V III () N 30,MLN, Test, PO L mask
Number of epochs 0 e , Batch size 0 e TRAIN NETWORK	g, 3-0 = 50, 3-0 = 150 g, 3-0 = 12, 3-0 = 8 PREDICT LABELS	Videos La Windows (C.) Lo DATA (D:) File name: 30_MLN, Test, ROL_mask	✓ ("tiff) ✓ Open Cancel

- Click 'Predict Labels' to generate the label-free probability images

- Progress can be tracked in the Command Prompt console

- Figure 1 demonstrates probability images for each class (central z-plane for 3-D data)

- **Outputs** are saved to the Desktop ready for loading into the CellProfiler pipelines enabling cell feature extraction (described below).

~					
1				Figure 2	- 🗆 X
Recycle Bin		I 🖸 🧧 🖛 Standalene_software_biostudie	s	File	
💽 МАПАВ Арр	- 🗆 ×	File Home Share View			
Train a label-free segmentation model	Generate label-free predictions	🖈 🗎 📋 🅉 Cut	📕 🖡 🗙 📑 📕	Central zplane: example net	work predictions
Load reflectance image:	Load pretrained model:	access Paste Paste shortcut	to to folder	LF-nuclei LF-actin	Background/other
Image file	MLN_Biostudies_27-May-2021-1355 mat	Clipboard	Organise		
Image x size 0	•	$\leftarrow \rightarrow \land \uparrow \square \rightarrow$ This PC \rightarrow DATA (D:) :	John > 2021 > Standalone_software_b		
image y star 0	Pretrained model information:	Poruments 0	Name	A CONTRACTOR OF LAND	總
Image z nize 0	Number of epochs 150	Downloads			
Jabel_free	Datch s be	h Music	2 WINDOWS10 INSTALL label free		
ediction		E Pictures	3 Example pretrained models		
Nuclear label file		Videos	4_2D_CellProfiler_pipeline	Uncertainty	
Load cell outline labels	Load reflectance image:	L Windows (C:)	5_3D_CellProfiler_pipeline		
Cell outline labels file	Image file 3D_MLN_Test_reflectance tilf	DATA (D:)	¢		12
I nari barkrimuntiniher labele:	Image x size 1024	5 items 1 item selected			截
Background labels file	image y size 1024	LabelFreeStainPrediction			1. A A A A A A A A A A A A A A A A A A A
Land POL mark // maniparty	Image z size 107	Starting label-free stain predicti	an		
		starting 3-D block-wise processing inished.	of the reflectance data		
NOT Mask the					
				(
	Load BOI-mask (if required):				
Number of epochs 0 8 g, 2-0 = 50; 3-0 = 150	BOI mask file 3D MLN Test BOI				
Batch size 0 e.g., 2-D = 12; 3-D = 8.					
and the second se					
TRAIN NETWORK	PREDICT LABELS				
		L			

Windows 10: Running the label-free cell segmentation deep learning scripts on an NVIDIA GPU using MATLAB R2021b and the Deep Learning Toolbox

Running these deep learning scripts in MATLAB requires:

- MATLAB (version R2019a or later)
- Deep Learning Toolbox
- Image Processing Toolbox
- Computer Vision Toolbox

Installation Steps: 1. Install MATLAB

Download MATLAB Simulink Str x +

https://ch.mathworks.com/downloads/web_downloads

- Login to your MathWorks account to access the downloads page at the link above.
- Download and run the installer for MATLAB R2021b.

	*	Ividulivv01KS°					
	Dov	wnloads					
	FAQ	Installation and Licen	nsing Help				C Cc
	Selec	t Release	P 2021h				About R2021b
	✓ R	2021b	120210				Latest Features
	R	2021a	Get MATLAB and	d Simulink Products			Release Notes
	R	2020b					System Requirements
				Download for V (196 MB) Includes R2021	Windows		
				(
			Installation Instru	uctions			
			1. Download and	d launch the installer.			
			2. Sign in as jw2	2020@cam.ac.uk.			
			In addition to in download produ	stalling MATLAB, you can also a uct files for installation on an of	add products to your ex ffline machine.	xisting installation or	
			Download details:			0	
			 matian R20210 		001-1-(71-10-LL00C01		
orks Product Installer	01			win64.exe (196 MB; MD5: 299be17c3	X		
orks Product Installer		PRODUCTS	OPTIONS	win64 exe (196 MB; MD5: 299be17c3	× • Du Con	uring instanputer Vis	llation, install sion, Deep Lea
ILCENSING		PRODUCTS	OPTIONS	Advanced Options CONFIRMATION	× • Du Con and	uring insta nputer Vis I Image Pr	illation, install sion, Deep Lea ocessing Tool
erks Product Installer	216 DESTINATION	PRODUCTS O ed products are	OPTIONS OPTIONS	Advanced Options CONFIRMATION	 - Du Con and by s 'Pro 	uring insta nputer Vis Image Pr selecting t oducts' ste	allation, install sion, Deep Lea ocessing Tool hem at the sp of the
orks Product Installer	DESTINATION	PRODUCTS O ed products are	OPTIONS OPTIONS	Advanced Options CONFIRMATION	- Du Com and by s 'Pro inst	uring instanputer Vis Image Pr Selecting t oducts' ste	allation, install sion, Deep Lea ocessing Tool hem at the sp of the
Cortes Product Installer Cortes Product In	215 DESTINATION Products (recommend Select All Bioinformatics Toolbox	PRODUCTS O ed products are	OPTIONS OPTIONS	Advanced Options CONFIRMATION	× - Du Com and by s 'Pro inst	uring insta nputer Vis Image Pr selecting t oducts' ste callation.	allation, install sion, Deep Lea ocessing Tool hem at the p of the
Corris Product Installer	21 DESTINATION Products (recommend Select All Bioinformatics Toolbox Communications Toolbox	PRODUCTS O ed products are	OPTIONS O e preselected)	Advanced Options CONFIRMATION	× - Du Con and by s 'Pro inst	uring insta nputer Vis Image Pr selecting t oducts' ste callation.	allation, install sion, Deep Lea ocessing Tool hem at the sp of the
LICENSING	DESTINATION Products (recommend Select All Bioinformatics Toolbox Computer Vision Toolbox	PRODUCTS O ed products are	options options e preselected)	Advanced Options CONFIRMATION	× - Du Con and by s 'Pro inst	uring instan nputer Vis Image Pr selecting t oducts' ste callation.	allation, install sion, Deep Lea ocessing Tool hem at the p of the stallation is
LICENSING	DESTINATION Products (recommend Select All Bioinformatics Toolbox Computer Vision Toolbox Control System Toolbox	PRODUCTS O ed products are	OPTIONS e preselected)	Advanced Options CONFIRMATION	 - Du Con and by s 'Pro inst - Or 	uring insta nputer Vis Image Pr selecting t oducts' ste callation.	allation, install sion, Deep Lea ocessing Tool hem at the sp of the stallation is
ICENSING	DESTINATION DESTINATION Products (recommend Select All Bioinformatics Toolbox Computer Vision Toolbox Computer Vision Toolbox Control System Toolbox Control System Toolbox	PRODUCTS O ed products are	OPTIONS OPTIONS	Advanced Options CONFIRMATION	 - Du Con and by s 'Pro inst - Or com 	Iring instanputer Vis Image Pr selecting to oducts' ste callation. nce the ins pplete, you	allation, install sion, Deep Lea ocessing Tool hem at the ep of the stallation is a can proceed
Cortes Product Installer	2110 DESTINATION Computer State Compared Select All Bioinformatics Toolbox Computer Vision Toolbox Control System Toolbox Control System Toolbox Curve Fitting Toolbox Database Toolbox	PRODUCTS O ed products are	OPTIONS O e preselected)	Advanced Options CONFIRMATION	× - Du Com and by s 'Pro inst - Or com	uring instanputer Vis Image Pr selecting to oducts' ste callation. nce the ins nplete, you	allation, install sion, Deep Lea ocessing Tool hem at the ep of the stallation is a can proceed cripts
LICENSING	210 DESTINATION C products (recommend Select All Bioinformatics Toolbox Computer Vision Toolbox Control System Toolbox Control System Toolbox Control System Toolbox Data Acquisition Toolbox Database Toolbox Database Toolbox	PRODUCTS O ed products are	options e preselected)	Advanced Options CONFIRMATION	× - Du Com and by s 'Pro inst - Or com	uring insta nputer Vis Image Pr selecting t oducts' ste callation. nce the ins nplete, you ning the s	allation, install sion, Deep Lea ocessing Tool hem at the ep of the stallation is a can proceed cripts.
Criss Product Installer	DESTINATION DESTINATION Products (recommend Select All Select All Select All Communications Toolbox Computer Vision Toolbox Computer Vision Toolbox Control System Toolbox Data Acquisition Toolbox Database Toolbox Database Toolbox Database Toolbox DDS Blockset	PRODUCTS ed products are	options o e preselected)	Advanced Options CONFIRMATION	× - Du Com and by s 'Pro inst - Or com run	uring instanputer Vis Image Pr selecting t oducts' ste callation. nce the ins nplete, you ning the s	allation, install sion, Deep Lea ocessing Tool hem at the p of the stallation is a can proceed cripts.
criss Product Installer	DESTINATION DESTINATION Products (recommend Select All Bioinformatics Toolbox Communications Toolbox Computer Vision Toolbox Control System Toolbox Courve Fitting Toolbox Curve Fitting Toolbox Data Acquisition Toolbox Database Toolbox Database Toolbox Database Toolbox Database Toolbox Database Toolbox DDS Blockset Deep Learning HDL Toolbox	PRODUCTS ed products are	options options e preselected)	Advanced Options CONFIRMATION	× - Du Com and by s 'Pro inst - Or com run	Iring insta nputer Vis Image Pr selecting t oducts' ste callation. nce the ins nplete, you ning the s	allation, install sion, Deep Lea ocessing Tool hem at the p of the stallation is a can proceed cripts.
Contract Installer	DESTINATION DESTINATION Products (recommend Select All Bioinformatics Toolbox Computer Vision Toolbox Computer Vision Toolbox Control System Toolbox Curve Fitting Toolbox Data Acquisition Toolbox DataAcquisition Toolbox DataAcquisition Toolbox DataAcquisition Toolbox DataAcquisition Toolbox DataAcquisition Toolbox Deep Learning Toolbox	PRODUCTS O ed products are	options e preselected)	Advanced Options CONFIRMATION	 - Du Con and by s 'Pro inst - Or com run 	Iring insta nputer Vis Image Pr selecting t oducts' ste callation. nce the ins nplete, you ning the s	allation, install sion, Deep Lea ocessing Tool hem at the ep of the stallation is a can proceed cripts.
Contract Installer	DESTINATION DESTINATION Products (recommend Select All Bioinformatics Toolbox Computer Vision Toolbox Control System Toolbox Control System Toolbox Control System Toolbox Control System Toolbox Data Acquisition Toolbox Data Deep Learning HDL Toolbox Deep Learning Toolbox	PRODUCTS O ed products are	options e preselected)	Advanced Options CONFIRMATION	× - Du Con and by s 'Prc inst - Or com	Iring insta nputer Vis Image Pr selecting t oducts' ste callation. nce the ins nplete, you ning the s	allation, install sion, Deep Lea ocessing Tool hem at the ep of the stallation is a can proceed cripts.
orks Product Installer	DESTINATION DESTINATION Products (recommend Select All Bioinformatics Toolbox Computer Vision Toolbox Computer Vision Toolbox Control System Toolbox Course Fitting Toolbox DataAscquistion Toolbox DataAscquistion Toolbox DataAscquistion Toolbox DataAscquistion Toolbox DataAscquistion Toolbox DataAscquistion Toolbox DataAscquistion Toolbox DataBase Toolbox DataBase Toolbox DataBase Toolbox DataBase Toolbox Deep Learning HDL Toolbox	PRODUCTS O ed products are	options o e preselected)	Advanced Options CONFIRMATION	× - Du Con and by s 'Pro inst - Or com run	Iring insta nputer Vis Image Pr selecting t oducts' ste callation. nce the ins nplete, you ning the s	allation, install sion, Deep Lea ocessing Tool hem at the ep of the stallation is a can proceed cripts.
2. Training a 2-D UNET Model

-A screencast video is included with the BioStudies archive.

- Download and unzip the MATLAB BioStudies project archive at a suitable location on your computer.



- Inside the MATLAB_2D_UNET folder, there is a sequentially-organised workflow starting with raw data and ending with a CellProfiler pipeline that obtains cell features from the label-free cell segmentation:



- To train a 2-D UNET model, open MATLAB and set the working directory to the "2_TRAIN_Unet" directory. Open the "A_TRAIN_UNET" script.

MATLAB R2021	b - acade	mic use	1											
HOME	PLOT		APPS											
New New Script Live Script	New FillE	Open	G Find Files	Import Data	Save Workspace VA	B New Variable Image: the state of the st	Favorites	Analyze Code	Layout	 Preferences Set Path Parallel ENVIRONMENT 	Add-Ons	? Help	Community Request Support Learn MATLAB RESOURCES	
💠 🌩 🛅 🖾 🎘	•	D: •	John 🕨 2021 🕨	MATLA	B_biostudies	MATLAB_2D_UNET +	2_TRAIN_U	net 🕨						
Command Window	N										\odot	Worksp	ace	
fx >>												Name	A	Value

- Update the path to the training data (located in the "1_ImageData" folder).

- Update the path to tissue ROI mask which isolates the lymphoid follicle from the surrounding tissue (located in the "1_ImageData" folder).

- Update the path to the pixel classification labels (located in the "1_ImageData" folder)

D:\	Jonn	021\MAILAB_biostudies\N	ATLAB_2D_UNET\2_TRAIN_U	Inet\A_IKAIN_UNEI.m								
ED	ITOR	PUBLISH	VIEW									
New V	Open	Save FILE Compare ▼	Go To WAVIGATE	Refactor CODE	Analyze	Run Section & Ri Sector	ection Break un and Advance un to End TION	Run	Step RUN	Stop		
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18		FILE Clear close all clc % Read in the ref [raining_data = i % Read in the mas hask = imread('D: mask = uintlo(imp % Apply the mask Training_data = T % Rescale the tra Training_data = d % Read in the pix	NAVIGATE ilectance data for t mread('D:\John\2021 k to isolated just \John\2021\MATLAB_b inarize(mask)); to the training dat raining_data .* mas shining data in the i louble(1*mat2gray(Tr tel-class_labels_cres	CODE raining \MATLAB_biostudie the lymphoid tiss iostudies\MATLAB_ a k ; nterval [0 1] aining_data, [0 40 ated from the puc	ANALYZE s\MATLAB_2D_ ue 2D_UNET\1_Im 295])) ; lei and acti	SECT UNET\1_Imi HageData\TH	ION IgeData\TRAIN KAIN_IMAGE.tii	_IMAGE.	RUN	; % c	nannel 3 is re	flectance
19 20 21 22 23 24 25 26 27 28 29		<pre>Training_labels = % Visually inspec figure(1) subplot(1,2,1) imshow(Training_d title(`Reflectanc subplot(1,2,2) imshow(Training_l title(`Pixel class</pre>	<pre>imread('D:\John\20 t the data and labe lata, []) e data') labels, []) sification labels')</pre>	21\MATLAB_biostud ls	ies\MATLAB_2	2D_UNET\1_1	imageData\TRA1	IN_LABE	LS.png	');	J	

- On Line 46, update the path to the pixel classification labels (located in the "1_ImageData" folder) used by the "pixelLabelDatastore" function.

D:\John\2021\MATLAB_biostudies\MATLAB_2D_UNET\2_TRAIN_Unet\A_TRAIN_UNET.m EDITOR fx % % % Section Break ¢ 2 🕂 🛅 🔚 😥 Compare 🕶 D Profiler Refactor Run and Advance Q Find -Analyze Run Section Run to End New Open Save 🚔 Print 🕶 Go To Run Step Stop Bookmark 💌 -fi -NAVIGATE ANALYZE SECTION FILE CODE RUN %% Create 'Datastores' for the reflectance information and matching pixel-class labels 36 % Read saved mat file containing reflectance info into an image datastore |
% Directions for processing the mat file-type are in the accompanying 'matReader' function
reflectance_ds = imageDatastore([pwd, '/mat_training_data/'],'FileExtensions','.mat','ReadFcn',@matReader); 37 38 39 40 41 % Read pixel labels into a pixel label datastore 42 PixelClassNames = ["LF_actin","LF_nuclei","BackgroundOther"]; 43 PixelLabel_ds = pixelLabelDatastore('D:\John\2021\MATLAB_biostudies\MATLAB_2D_UNET\1_ImageData\TRAIN_LABELS.png', PixelClassNames,Label_Ids); 44 45 46

- At the bottom of the script on Line 195, update the path to the saved_network_directory. - This should be folder 3 of the workflow ("3_Saved_models").



- Run the script by clicking the green arrow at the top.

- Model training takes several hours (~ 4h on a NVIDIA 1080 Ti GPU)

- The newly-trained model will be saved in the "3_Saved_models" directory with a new time/date stamp.

D:\John\2021\MATLAB_biostudies\MATLAB_2D_UNET\2_TRAIN_Unet\A_TRAIN_UNET.m

ED	ITOR		PUBLISH	VIEW									
New	Open	Save	🔄 Compare 👻 🚍 Print 💌	Go To	♀ ↔ Q Find ▼ Bookmark ▼	Refactor	% ‰ %7 ▶ +3 ₽ •4 •	😓 Profiler 🚽 Analyze	Run Section	E Section Break Run and Advance Run to End	Run	G Step	Stop
		FILE			NAVIGATE	C	ODE	ANALYZE		SECTION		RUN	

3. Testing a pretrained 2-D model using unseen data

-A screencast video is included with the BioStudies archive.

- Change the MATLAB working directory to the "4_Test_Unet" directory. Open the "A_TEST_UNET" script.



- Change the path to the unseen test image (located in the "1_ImageData" folder).

- Specify the ROI mask which identifies the lymphoid tissue (located in the "1_ImageData" folder).

- Specify which pretrained network to use (pretrained networks are located in the "3_Saved_models" folder.

D:\John\2021\MATLAB_biostudies\MATLAB_2D_UNET\4_TEST_Unet\A_TEST_UNET.m

E	DITOR		PUBLISH	VIEW					
New	Open	Save	E Compare ▼ ➡ Print ▼	Go To AVIGATE	Refactor	Profiler	Run Section Break	Run Step	Stop
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18		FILE clear close clc % Load Mask · Mask · % Resc TEST_I % Masl TEST_I % Load	all <u>d the reflect</u> DATA = imread d the mask fo = imread('D:\ = double(imbi <u>cale the test</u> DATA_rescaled ATA_rescaled ATA_rescaled	<pre>naviGATE ance data from the ('D:\John\2021\MATLAB_bi narize(Mask)); data [0 1] data [0 1] data d data = TEST_DATA_rescal d natwork</pre>	unseen test image AB_biostudies\MAT ue region ostudies\MATLAB_2 y(TEST_DATA, [0 4 ed .* Mask ;	ANALYZE	SECTION T\1_ImageData\TEST_IMAG mageData\TEST_IMAGE.ti	RUN	3 is reflectance channel
19 20	l	load('D:\John\2021	\MATLAB_biostudies	MATLAB_2D_UNET\3_	Saved_mode	ls\TEST_e5023-May-2021	-23-28-02.mat');

- Click the "Run" button to process the unseen reflectance data with the selected pretrained network:



- Probability maps for the LF-nuclei, LF-actin and Background/Other classes will be saved in the working directory named ready for input into the CellProfiler pipeline (filenames Ch_5_lm_001, Ch_6_lm_001, Ch_7_lm_001, respectively).

\leftarrow \rightarrow \checkmark \uparrow MATLAB_biostudies \rightarrow	MATI	AB_2D_UNET > 4_TEST_Unet	م ت ۲	Search 4_TEST_Unet
Documents	^	Name	Date	Туре
🕹 Downloads		A_TEST_UNET	30/11/2021 14:08	MATLAB Code
b Music		Ch_5_lm_001	30/11/2021 14:33	TIFF File
Pictures		Ch_6_lm_001	30/11/2021 14:33	TIFF File
Videos		Ch_7_lm_001	30/11/2021 14:33	TIFF File

4. Training a 3-D UNET Model

-A screencast video is included with the BioStudies archive.

- Inside the MATLAB_3D_UNET folder, there is a sequentially-organised workflow starting with raw data and ending with a CellProfiler pipeline that obtains 3-D cell features from the label-free cell segmentation:

2021	> MATLAB_biostudies > MATLAB_3D_UNET	5 V	
^	Name	Date modified	Туре
	📙 1_ImageData	30/11/2021 14:08	File folder
	2_TRAIN_Unet	30/11/2021 14:12	File folder
	3_Saved_models	30/11/2021 14:10	File folder
	4_TEST_Unet	30/11/2021 14:12	File folder
	5_CellProfiler_cell_measurements	30/11/2021 14:10	File folder

- To train a 3-D UNET model, open MATLAB and set the working directory to the "2_TRAIN_Unet" directory. Open the "A_TRAIN_3D_UNET" script.



- Specify the path to the bioformats library. This is included inside the "2_TRAIN_Unet" folder ("bfmatlab").

- Update the path to the training data directory (located in the "1_ImageData/TRAIN/DATA" folder).

- Specify the channel number in the training data that contains the reflectance information (here, '3').

- Specify the location of the pixel classification training labels (located at "1_ImageData/TRAIN/LABELS" folder.



- Specify the location to save the model when training completes. This should be the "3_Saved_models" directory.

227	%% Train and save the network			
228	close all			
229				
230	% Specify location to save the network			
231	<pre>saved_network_directory = 'D:\John\2021\MATLAB_biostudies\MATLAB_3D_UNET\3_</pre>	Saved_n	<pre>nodels\';</pre>	
232				
233	% Train the network			
234	<pre>modelDateTime = datestr(now,'dd-mmm-yyyy-HHMM');</pre>			
235	<pre>[net,info] = trainNetwork(Training_ds,lgraph,options);</pre>			
236				
237	% Timestamp and save the network after training			
238	<pre>save([saved_network_directory,'MLN_Biostudies_',modelDateTime,'.mat'],'net'</pre>	, 'optic	ons','inf	o');
239				

- Click "Run" to commence model training.

- This takes considerable time (approximately 6h on a NVIDIA 1080Ti GPU card).



6. Testing a pretrained 3-D model using unseen data

-A screencast video is included with the BioStudies archive.

- Change the MATLAB working directory to the "4_Test_Unet" directory. Open the "A_TEST_3D_UNET" script.



- Specify the path to the Bioformats library. (This is included inside the "4_TEST_Unet" folder ("bfmatlab").

- Specify the location of the unseen test image-data. This is located inside the "1_ImageData" folder at "1_ImageData/TEST/".



- Specify a pretrained 3-D Unet model from the "3_Saved_models" directory.

40	%%
41	% load the trained 3D Unet network
42	load('D:\John\2021\MATLAB biostudies\MATLAB 3D UNET\3 Saved models\MLN Biostudies 27-May-2021-1355.mat);
43	

- Click "Run" to process the unseen reflectance data with the 3-D Unet model.

🗐 D:	\John\2)21\MA	TLAB_biostudies\M	IATLAB_	D_UNET\4_TEST_Un	et\A_TEST_	3D_UNET.m						
E	DITOR		PUBLISH	VIEW	1							6 4	1
New	Open	Save	🔄 Compare 👻 🚍 Print 👻	Go To		Refactor	% ‰ %7 ∑ € ⊡	🕤 Profiler 📝 Analyze	Run Section	 Section Break Run and Advance Run to End 	Run	G Step	Stop
4.20		FILE			NAVIGATE	0	ODE	ANALYZE		SECTION		RUN	

- Probability maps for the LF-nuclei, LF-actin and Background/Other classes will be saved as multipage .TIFF files in the working directory named ready for input into the CellProfiler pipeline (filenames Ch_3_lm_001, Ch_4_lm_001, Ch_5_lm_001, respectively).



```
% MATLAB SCRIPT: TRAIN 2D UNET
% All code, image-data and screen-cast tutorial videos available for download at
% the Biostudies database under accession number S-BSST742
clear
close all
clc
% Read in the reflectance data for training
Training data =
imread('D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\1 ImageData\TRAIN IMAGE.ti
f',3) ; % channel 3 is reflectance
% Read in the mask to isolated just the lymphoid tissue
mask =
imread('D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\1 ImageData\TRAIN IMAGE.ti
f',4);
mask = uint16(imbinarize(mask));
% Apply the mask to the training data
Training data = Training data .* mask ;
% Rescale the training data in the interval [0 1]
Training data = double(1*mat2gray(Training data, [0 4095])) ;
% Read in the pixel-class labels created from the nuclei and actin staining
Training labels =
imread('D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\1 ImageData\TRAIN LABELS.p
ng');
% Visually inspect the data and labels
figure(1)
subplot(1,2,1)
imshow(Training data, [])
title('Reflectance data')
subplot(1,2,2)
imshow(Training labels, [])
title('Pixel classification labels')
%% Once happy with labels and data, commit the reflectance information data to
sub-directory in .mat format
if ~exist('mat training data', 'dir')
   mkdir('mat training data');
end
save([pwd,'/mat training data/','Training data.mat'],'Training data');
%% Create 'Datastores' for the reflectance information and matching pixel-class
labels
% Read saved mat file containing reflectance info into an image datastore
% Directions for processing the mat file-type are in the accompanying
'matReader' function
reflectance ds = imageDatastore([pwd,
'/mat training data/'],'FileExtensions','.mat','ReadFcn',@matReader) ;
% Read pixel labels into a pixel label datastore
PixelClassNames = ["LF_actin","LF_nuclei","BackgroundOther"];
Label_Ids = 1:3; % these represent the pixel values in the labels png file
PixelLabel ds =
pixelLabelDatastore('D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\1 ImageData\T
RAIN LABELS.png',...
```

PixelClassNames,Label_Ids);

```
%% Define augmentation options
augmenter = imageDataAugmenter(...
                                             %%% description and defaults %%%
    'FillValue',0,...
                                      % define out-of-bounds points when resampling
0
    'RandXReflection', true,...
                                      % Random reflection in the left-right
direction false
    'RandYReflection',true,...
                                      % Random reflection in the top-bottom
direction false
    'RandRotation',[0 360],...
                                      % Range of rotation, in degrees [0 0]
    'RandScale',[1 1],...
                                      % Range of uniform (isotropic) scaling [1 1]
    'RandXScale',[1 1],...
'RandYScale',[1 1],...
                                      % Range of horizontal scaling [1 1]
                                      % Range of vertical scaling [1 1]
                                      % Range of horizontal shear [0 0]
    'RandXShear',[0 0],...
    'RandYShear',[0 0],...
                                      % Range of vertical shear [0 0]
    'RandXTranslation',[0 0],...
                                    % Range of horizontal translation [0 0]
    'RandYTranslation',[0 0]...
                                     % Range of vertical translation [0 0]
    );
%% Create a training datastore comprising matching patches (256x256 pixels) of
image-data and training labels
training ds =
randomPatchExtractionDatastore(reflectance ds,PixelLabel ds,[256,256],...
     'PatchesPerImage',12000,'DataAugmentation',augmenter); % 743 draws == 1
epoch
%% CREATE THE UNET
% input layer 256x256x1
% encoder depth 4
% 64 filters at the level of the first encoder
lgraph = layerGraph();
tempLayers = [
    imageInputLayer([256 256 1], "Name", "ImageInputLayer")
    convolution2dLayer([3 3],64, "Name", "Encoder-Stage-1-Conv-
1","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Encoder-Stage-1-ReLU-1")
    convolution2dLayer([3 3],64,"Name","Encoder-Stage-1-Conv-
2","Padding","same","WeightsInitializer","he")
    reluLayer("Name","Encoder-Stage-1-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    maxPooling2dLayer([2 2],"Name","Encoder-Stage-1-MaxPool","Stride",[2 2])
    convolution2dLayer([3 3],128,"Name","Encoder-Stage-2-Conv-
1", "Padding", "same", "WeightsInitializer", "he")
    reluLayer("Name","Encoder-Stage-2-ReLU-1")
convolution2dLayer([3 3],128,"Name","Encoder-Stage-2-Conv-
2","Padding","same","WeightsInitializer","he")
reluLayer("Name","Encoder-Stage-2-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    maxPooling2dLayer([2 2],"Name","Encoder-Stage-2-MaxPool","Stride",[2 2])
    convolution2dLayer([3 3],256,"Name","Encoder-Stage-3-Conv-
1","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Encoder-Stage-3-ReLU-1")
    convolution2dLayer([3 3],256,"Name","Encoder-Stage-3-Conv-
2","Padding","same","WeightsInitializer","he")
```

```
reluLayer("Name", "Encoder-Stage-3-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    maxPooling2dLayer([2 2], "Name", "Encoder-Stage-3-MaxPool", "Stride", [2 2])
convolution2dLayer([3 3],512,"Name","Encoder-Stage-4-Conv-
1","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Encoder-Stage-4-ReLU-1")
    convolution2dLayer([3 3],512,"Name","Encoder-Stage-4-Conv-
2", "Padding", "same", "WeightsInitializer", "he")
    reluLayer("Name", "Encoder-Stage-4-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    dropoutLayer(0.5, "Name", "Encoder-Stage-4-DropOut")
    maxPooling2dLayer([2 2], "Name", "Encoder-Stage-4-MaxPool", "Stride", [2 2])
convolution2dLayer([3 3],1024,"Name","Bridge-Conv-
1","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Bridge-ReLU-1")
convolution2dLayer([3 3],1024,"Name","Bridge-Conv-
2","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Bridge-ReLU-2")
    dropoutLayer(0.5, "Name", "Bridge-DropOut")
transposedConv2dLayer([2 2],512,"Name","Decoder-Stage-1-
UpConv","BiasLearnRateFactor",2,"Stride",[2 2],"WeightsInitializer","he")
    reluLayer("Name", "Decoder-Stage-1-UpReLU")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    depthConcatenationLayer(2, "Name", "Decoder-Stage-1-DepthConcatenation")
    convolution2dLayer([3 3],512,"Name","Decoder-Stage-1-Conv-
1","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Decoder-Stage-1-ReLU-1")
convolution2dLayer([3 3],512, "Name", "Decoder-Stage-1-Conv-
2", "Padding", "same", "WeightsInitializer", "he")
    reluLayer("Name", "Decoder-Stage-1-ReLU-2")
    transposedConv2dLayer([2 2],256,"Name","Decoder-Stage-2-
UpConv", "BiasLearnRateFactor", 2, "Stride", [2 2], "WeightsInitializer", "he")
    reluLayer("Name", "Decoder-Stage-2-UpReLU")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    depthConcatenationLayer(2, "Name", "Decoder-Stage-2-DepthConcatenation")
    convolution2dLayer([3 3],256,"Name","Decoder-Stage-2-Conv-
1","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Decoder-Stage-2-ReLU-1")
convolution2dLayer([3 3],256, "Name", "Decoder-Stage-2-Conv-
2", "Padding", "same", "WeightsInitializer", "he")
    reluLayer("Name", "Decoder-Stage-2-ReLU-2")
transposedConv2dLayer([2 2],128,"Name","Decoder-Stage-3-
UpConv","BiasLearnRateFactor",2,"Stride",[2 2],"WeightsInitializer","he")
    reluLayer("Name", "Decoder-Stage-3-UpReLU")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    depthConcatenationLayer(2, "Name", "Decoder-Stage-3-DepthConcatenation")
    convolution2dLayer([3 3],128,"Name","Decoder-Stage-3-Conv-
1", "Padding", "same", "WeightsInitializer", "he")
    reluLayer("Name", "Decoder-Stage-3-ReLU-1")
```

```
convolution2dLayer([3 3],128,"Name","Decoder-Stage-3-Conv-
2","Padding","same","WeightsInitializer","he")
    reluLayer("Name", "Decoder-Stage-3-ReLU-2")
    transposedConv2dLayer([2 2],64,"Name","Decoder-Stage-4-
UpConv","BiasLearnRateFactor",2,"Stride",[2 2],"WeightsInitializer","he")
    reluLayer("Name", "Decoder-Stage-4-UpReLU")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    depthConcatenationLayer(2, "Name", "Decoder-Stage-4-DepthConcatenation")
    convolution2dLayer([3 3],64,"Name","Decoder-Stage-4-Conv-
1", "Padding", "same", "WeightsInitializer", "he")
    reluLayer("Name", "Decoder-Stage-4-ReLU-1")
    convolution2dLayer([3 3],64, "Name", "Decoder-Stage-4-Conv-
2","Padding","same","WeightsInitializer","he")
    reluLayer("Name","Decoder-Stage-4-ReLU-2")
    convolution2dLayer([1 1],3,"Name","Final-
ConvolutionLayer", "Padding", "same", "WeightsInitializer", "he")
    softmaxLayer("Name", "Softmax-Layer")
    pixelClassificationLayer("Name", "Segmentation-Layer")];
lgraph = addLayers(lgraph,tempLayers);
clear tempLayers;
% encoder / decoder connections
lgraph = connectLayers(lgraph, "Encoder-Stage-1-ReLU-2", "Encoder-Stage-1-
MaxPool");
lgraph = connectLayers(lgraph,"Encoder-Stage-1-ReLU-2","Decoder-Stage-4-
DepthConcatenation/in2");
lgraph = connectLayers(lgraph, "Encoder-Stage-2-ReLU-2", "Encoder-Stage-2-
MaxPool");
lgraph = connectLayers(lgraph, "Encoder-Stage-2-ReLU-2", "Decoder-Stage-3-
DepthConcatenation/in2");
lgraph = connectLayers(lgraph, "Encoder-Stage-3-ReLU-2", "Encoder-Stage-3-
MaxPool");
lgraph = connectLayers(lgraph, "Encoder-Stage-3-ReLU-2", "Decoder-Stage-2-
DepthConcatenation/in2");
lgraph = connectLayers(lgraph,"Encoder-Stage-4-ReLU-2","Encoder-Stage-4-
DropOut");
lgraph = connectLayers(lgraph,"Encoder-Stage-4-ReLU-2","Decoder-Stage-1-
DepthConcatenation/in2");
lgraph = connectLayers(lgraph, "Decoder-Stage-1-UpReLU", "Decoder-Stage-1-
DepthConcatenation/in1");
lgraph = connectLayers(lgraph, "Decoder-Stage-2-UpReLU", "Decoder-Stage-2-
DepthConcatenation/in1");
lgraph = connectLayers(lgraph, "Decoder-Stage-3-UpReLU", "Decoder-Stage-3-
DepthConcatenation/in1");
lgraph = connectLayers(lgraph, "Decoder-Stage-4-UpReLU", "Decoder-Stage-4-
DepthConcatenation/in1");
% analyzeNetwork(lgraph) % comment in to check structure and errors
plot(lgraph) % comment in to show network structure
%% Specify training options
options = trainingOptions('sgdm',...
    'InitialLearnRate',0.05, ...
    'Momentum',0.9,...
    'L2Regularization',0.0001,...
    'MaxEpochs',50,...
    'MiniBatchSize',12,...
```

```
'LearnRateSchedule', 'piecewise',...
    'Shuffle', 'every-epoch',...
    'GradientThresholdMethod', 'l2norm',...
    'GradientThreshold',0.05, ...
    'Plots', 'training-progress', ...
    'VerboseFrequency',20,...
    'ExecutionEnvironment', 'auto');
%% Train and save the network
close all
% Specify location to save the network
saved network directory =
'D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\3 Saved models\';
% Train the network
modelDateTime = datestr(now, 'dd-mmm-yyyy-HH-MM-SS');
[net,info] = trainNetwork(training ds,lgraph,options);
% Timestamp and save the network after training
save([saved network directory,'PeyersPatchBiostudies ',modelDateTime,'.mat'],...
    'net','options','augmenter','info');
```



2-D Unet architecture schematic. The network uses an input layer for the reflectance data of 256x256x1 (x, y, channels). The best performing three-class Unet architecture uses an encoder depth of 4 with 64 filters at the level of the first encoder (shown, **Figure S2**). The network uses complete up-convolutional expansion to yield outputted probability maps that are identically sized to the input layer.

```
% MATLAB SCRIPT: TEST 2D UNET
% All code, image-data and screen-cast tutorial videos available for download at
% the Biostudies database under accession number S-BSST742
clear
close all
clc
% Load the reflectance data from the unseen test image
TEST DATA =
imread('D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\1 ImageData\TEST IMAGE.tif
',3); % 3 is reflectance channel
% Load the mask for the lymphoid tissue region
Mask =
imread('D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\1 ImageData\TEST IMAGE.tif
',4);
Mask = double(imbinarize(Mask)) ;
% Rescale the test data [0 1]
TEST DATA rescaled = double(1*mat2gray(TEST DATA, [0 4095])) ;
% Mask the rescaled data
TEST DATA rescaled = TEST DATA rescaled .* Mask ;
% Load a pretrained network
load('D:\John\2021\MATLAB biostudies\MATLAB 2D UNET\3 Saved models\Rescaled0-
1 2DUNET S7-RL-SA enc-4 filt-64 e50.mat');
%% Patch the reflectance data through the network
****
% Set the patch sizes to be passed to the net
patchSize = [2048 2048]; % decrease if insufficient GPU memory eg., [1024 1024]
% Segment blockwise then reassemble in full
% Define image dimensions
[height, width, nChannel] = size(TEST DATA rescaled);
patch = zeros([patchSize, nChannel], 'like', TEST_DATA_rescaled);
% Pad image to have dimensions as multiples of patchSize
padSize(1) = patchSize(1) - mod(height, patchSize(1));
padSize(2) = patchSize(2) - mod(width, patchSize(2));
im_pad = padarray (TEST_DATA_rescaled, padSize, 0, 'post');
[height_pad, width_pad, nChannel_pad] = size(im_pad);
% Preallocate some matrices to receive the network outputs
out Uncertainty Scores = zeros([size(im pad,1), size(im pad,2)], 'double');
out Pmap_cat1 = out Uncertainty Scores;
out Pmap cat2 = out Uncertainty Scores;
out_Pmap_cat3 = out_Uncertainty Scores;
% Loop through blocks of 'patchSize'
    for loop = 1:patchSize(1):height pad
        for j =1:patchSize(2):width pad
           for p = 1:nChannel
                patch(:,:,p) = squeeze( im_pad( loop:loop+patchSize(1)-1,...
                                               j:j+patchSize(2)-1,p));
           end
            % deploy net
```

```
[patch_seg, Scores, allScores] = semanticseg(patch, net,
'OutputType', 'double',...
                'ExecutionEnvironment', 'auto');
            % catch what comes out
            out Uncertainty Scores(loop:loop+patchSize(1)-1, j:j+patchSize(2)-1)
= Scores;
            out Pmap cat1(loop:loop+patchSize(1)-1, j:j+patchSize(2)-1) =
allScores(:,:,1);
            out_Pmap_cat2(loop:loop+patchSize(1)-1, j:j+patchSize(2)-1) =
allScores(:,:,2);
            out Pmap cat3(loop:loop+patchSize(1)-1, j:j+patchSize(2)-1) =
allScores(:,:,3);
        end
    end
% Remove padding from the network outputs
out Uncertainty Scores = out Uncertainty Scores(1:height, 1:width);
out_Pmap_cat1 = out_Pmap_cat1(1:height, 1:width);
out_Pmap_cat2 = out_Pmap_cat2(1:height, 1:width);
out_Pmap_cat3 = out_Pmap_cat3(1:height, 1:width);
%% visualise network outputs
figure(2)
clf(figure(2))
ax1 = subplot(2,3,1);
imshow(out Pmap_cat1,[])
title('LF-actin')
ax2 = subplot(2,3,2);
imshow(out_Pmap_cat2,[])
title('LF-nuclei')
ax3 = subplot(2,3,3);
imshow(out_Pmap_cat3,[])
title('Background/other')
ax4 = subplot(2,3,5);
imshow(out Uncertainty Scores,[])
title('Uncertainty')
linkaxes([ax1 ax2 ax3 ax4], 'xy')
%% Map to 16-bit and save for loading into CellProfiler
ui16_PMAP_cat1_xy = uint16(65535*mat2gray(out_Pmap_cat1, [0 1])) ;
imwrite(ui16_PMAP_cat1_xy, ['Ch_6_Im_001','.tiff']) ;
uil6 PMAP cat2 xy = uint16(65535*mat2gray(out_Pmap_cat2, [0 1]));
imwrite(ui16 PMAP cat2 xy, ['Ch 5 Im 001','.tiff']) ;
ui16 PMAP_cat3_xy = uint16(65535*mat2gray(out_Pmap_cat3, [0 1])) ;
imwrite(uil6 PMAP cat3_xy, ['Ch_7_Im_001','.tiff']) ;
```

```
% MATLAB SCRIPT: Train 3D UNET
% All code, image-data and screen-cast tutorial videos available for download at
% the Biostudies database under accession number S-BSST742
clear
close all
clc
% Specify path to bioformats library
addpath('D:\John\2021\MATLAB biostudies\MATLAB 3D UNET\2 TRAIN Unet\bfmatlab\')
% Specify directory containing training data
Training data directory =
'D:\John\2021\MATLAB_biostudies\MATLAB 3D UNET\1 ImageData\TRAIN\DATA\';
% Specify channel number in training data that contains reflectance information
RL training directory = 3;
% Specify directory containing training labels
Training labels path =
'D:\John\2021\MATLAB_biostudies\MATLAB_3D_UNET\1_ImageData\TRAIN\LABELS\';
8% COMMIT TRAINING DATA TO DIRECTORY rescaled [0 1] IN MAT FORMAT
if ~exist('mat training data', 'dir')
   mkdir('mat_training_data');
end
for loop = 1:2
   counter = sprintf('%03d',loop) ;
    % find metadata describing file
    reader = bfGetReader([Training data directory, 'TRAIN ',counter,'.tif']);
    omeMeta = reader.getMetadataStore();
    number of channels = omeMeta.getChannelCount(0);
    stackSizeZ = omeMeta.getPixelsSizeZ(0).getValue(); % number of Z slices
    % Load reflectance information
    for zplane = 1:stackSizeZ
        iPlane = reader.getIndex(zplane -1, RL_training_directory -1, 0) + 1; %
because zplanes and channels are numbered from zero
        channel zimage{zplane} = bfGetPlane(reader, iPlane);
    end
    IM DATA = cat(3,channel zimage{:}) ;
    IM DATA = double(1*mat2gray(IM DATA, [0 65535])) ;
    save([pwd,'/mat_training_data/','TRAIN_DATA_',counter,'.mat'], 'IM_DATA') ;
end
%% COMMIT TRAINING LABELS TO DIRECTORY IN MAT FORMAT
if ~exist('mat_training_labels', 'dir')
   mkdir('mat_training_labels');
end
for loop = 1:2
    counter = sprintf('%03d',loop) ;
        % load label information
        for zplane = 1:stackSizeZ
            iplane =
imread([Training_labels_path,'LABELS_',counter,'.tif'],zplane);
```

```
label zimage{zplane} = iplane ;
        end
        IM LABELS = cat(3,label zimage{:});
        IM LABELS = double(IM LABELS) ;
        save([pwd,'/mat_training_labels/','TRAIN_LABELS_',counter,'.mat'],
'IM LABELS') ;
end
% Once data is prepared for training, clear workspace
clear ;
%% Create 'Datastores' for the reflectance information and matching pixel-class
labels
% Read saved mat file containing reflectance info into an image datastore
% Directions for processing the mat file-type are in the accompanying 'matRead'
function
data location = [pwd, '\mat training data\'];
volReader = @(x) JWmatRead(x);
reflectance ds =
imageDatastore(data location, 'FileExtensions', '.mat', 'ReadFcn', volReader);
% Read pixel labels into a pixel label datastore
label location = [pwd, '\mat training labels\'];
labelReader = @(x) JWLabelRead(x);
classNames = ["LFNuclei", "LFActin", "BackgroundOther"];
pixelLabelID = 1:3; % these represent the pixel values in the labels file
PixelLabel_ds = pixelLabelDatastore(label_location,classNames,pixelLabelID,...
    'FileExtensions', '.mat', 'ReadFcn', labelReader);
%% Set up patch extraction from reflectance datastore
patchSize = [64 64 32];
patchPerImage = 375;
MiniBatchSize = 8; % set the batch size
reflectance patches ds =
randomPatchExtractionDatastore(reflectance ds, PixelLabel ds, patchSize, 'PatchesPe
rImage',patchPerImage);
reflectance patches ds.MiniBatchSize = MiniBatchSize;
% Augment the patches using 'augment3dPatch' function
Training ds = transform(reflectance patches ds,@JWaugment3dPatch);
%% CREATE THE 3D UNET
% input layer 64x64x32x1
% encoder depth 4
% 32 filters at the level of the first encoder
lgraph = layerGraph();
tempLayers = [
    image3dInputLayer([64 64 32 1], "Name", "ImageInputLayer")
    convolution3dLayer([3 3 3],32,"Name","Encoder-Stage-1-Conv-
1","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Encoder-Stage-1-BN-1")
    reluLayer("Name", "Encoder-Stage-1-ReLU-1")
convolution3dLayer([3 3 3],64,"Name","Encoder-Stage-1-Conv-
2","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Encoder-Stage-1-BN-2")
    reluLayer("Name", "Encoder-Stage-1-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
```

```
tempLayers = [
    maxPooling3dLayer([2 2 2], "Name", "Encoder-Stage-1-MaxPool", "Stride", [2 2 2])
convolution3dLayer([3 3 3],64,"Name","Encoder-Stage-2-Conv-
1","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Encoder-Stage-2-BN-1")
    reluLayer("Name", "Encoder-Stage-2-ReLU-1")
    convolution3dLayer([3 3 3],128,"Name","Encoder-Stage-2-Conv-
2", "Padding", "same", "WeightsInitializer", "he")
    batchNormalizationLayer("Name", "Encoder-Stage-2-BN-2")
    reluLayer("Name", "Encoder-Stage-2-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    maxPooling3dLayer([2 2 2], "Name", "Encoder-Stage-2-MaxPool", "Stride", [2 2 2])
convolution3dLayer([3 3 3],128,"Name","Encoder-Stage-3-Conv-
1","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Encoder-Stage-3-BN-1")
    reluLayer("Name", "Encoder-Stage-3-ReLU-1")
    convolution3dLayer([3 3 3],256,"Name","Encoder-Stage-3-Conv-
2","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Encoder-Stage-3-BN-2")
    reluLayer("Name", "Encoder-Stage-3-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    maxPooling3dLayer([2 2 2], "Name", "Encoder-Stage-3-MaxPool", "Stride", [2 2 2])
    convolution3dLayer([3 3 3],256,"Name","Encoder-Stage-4-Conv-
1", "Padding", "same", "WeightsInitializer", "he")
    batchNormalizationLayer("Name", "Encoder-Stage-4-BN-1")
    reluLayer("Name", "Encoder-Stage-4-ReLU-1")
    convolution3dLayer([3 3 3],512,"Name","Encoder-Stage-4-Conv-
2","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Encoder-Stage-4-BN-2")
    reluLayer("Name", "Encoder-Stage-4-ReLU-2")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    maxPooling3dLayer([2 2 2], "Name", "Encoder-Stage-4-MaxPool", "Stride", [2 2 2])
convolution3dLayer([3 3 3],512,"Name","Bridge-Conv-
1","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Bridge-BN-1")
    reluLayer("Name","Bridge-ReLU-1")
    convolution3dLayer([3 3 3],1024,"Name","Bridge-Conv-
2", "Padding", "same", "WeightsInitializer", "he")
    batchNormalizationLayer("Name", "Bridge-BN-2")
    reluLayer("Name", "Bridge-ReLU-2")
    transposedConv3dLayer([2 2 2],1024,"Name","Decoder-Stage-1-
UpConv", "BiasLearnRateFactor", 2, "Stride", [2 2 2], "WeightsInitializer", "he")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    concatenationLayer(4,2,"Name","Decoder-Stage-1-Concatenation")
    convolution3dLayer([3 3 3],512,"Name","Decoder-Stage-1-Conv-
1","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Decoder-Stage-1-BN-1")
    reluLayer("Name", "Decoder-Stage-1-ReLU-1")
    convolution3dLayer([3 3 3],512,"Name","Decoder-Stage-1-Conv-
2","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Decoder-Stage-1-BN-2")
```

```
reluLayer("Name", "Decoder-Stage-1-ReLU-2")
    transposedConv3dLayer([2 2 2],512,"Name","Decoder-Stage-2-
UpConv", "BiasLearnRateFactor", 2, "Stride", [2 2 2], "WeightsInitializer", "he")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    concatenationLayer(4,2,"Name","Decoder-Stage-2-Concatenation")
convolution3dLayer([3 3 3],256,"Name","Decoder-Stage-2-Conv-
1","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Decoder-Stage-2-BN-1")
    reluLayer("Name", "Decoder-Stage-2-ReLU-1")
    convolution3dLayer([3 3 3],256,"Name","Decoder-Stage-2-Conv-
2", "Padding", "same", "WeightsInitializer", "he")
    batchNormalizationLayer("Name", "Decoder-Stage-2-BN-2")
    reluLayer("Name", "Decoder-Stage-2-ReLU-2")
    transposedConv3dLayer([2 2 2],256,"Name","Decoder-Stage-3-
UpConv","BiasLearnRateFactor",2,"Stride",[2 2 2],"WeightsInitializer","he")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    concatenationLayer(4,2,"Name","Decoder-Stage-3-Concatenation")
    convolution3dLayer([3 3 3],128,"Name","Decoder-Stage-3-Conv-
1", "Padding", "same", "WeightsInitializer", "he")
    batchNormalizationLayer("Name", "Decoder-Stage-3-BN-1")
    reluLayer("Name", "Decoder-Stage-3-ReLU-1")
convolution3dLayer([3 3 3],128,"Name","Decoder-Stage-3-Conv-
2","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Decoder-Stage-3-BN-2")
    reluLayer("Name", "Decoder-Stage-3-ReLU-2")
    transposedConv3dLayer([2 2 2],128, "Name", "Decoder-Stage-4-
UpConv", "BiasLearnRateFactor", 2, "Stride", [2 2 2], "WeightsInitializer", "he")];
lgraph = addLayers(lgraph,tempLayers);
tempLayers = [
    concatenationLayer(4,2,"Name","Decoder-Stage-4-Concatenation")
convolution3dLayer([3 3 3],64,"Name","Decoder-Stage-4-Conv-
1","Padding","same","WeightsInitializer","he")
    batchNormalizationLayer("Name", "Decoder-Stage-4-BN-1")
    reluLayer("Name", "Decoder-Stage-4-ReLU-1")
convolution3dLayer([3 3 3],64, "Name", "Decoder-Stage-4-Conv-
2", "Padding", "same", "WeightsInitializer", "he")
    batchNormalizationLayer("Name", "Decoder-Stage-4-BN-2")
    reluLayer("Name", "Decoder-Stage-4-ReLU-2")
    convolution3dLayer([1 1 1],3,"Name","Final-
ConvolutionLayer", "Padding", "same", "WeightsInitializer", "he")
    softmaxLayer("Name", "Softmax-Layer")
    pixelClassificationLayer("Name", "Segmentation-Layer")];
lgraph = addLayers(lgraph,tempLayers);
clear tempLayers;
% encoder / decoder connections
lgraph = connectLayers(lgraph,"Encoder-Stage-1-ReLU-2","Encoder-Stage-1-
MaxPool");
lgraph = connectLayers(lgraph, "Encoder-Stage-1-ReLU-2", "Decoder-Stage-4-
Concatenation/in2");
lgraph = connectLayers(lgraph,"Encoder-Stage-2-ReLU-2","Encoder-Stage-2-
MaxPool");
lgraph = connectLayers(lgraph,"Encoder-Stage-2-ReLU-2","Decoder-Stage-3-
Concatenation/in2");
```

```
lgraph = connectLayers(lgraph, "Encoder-Stage-3-ReLU-2", "Encoder-Stage-3-
MaxPool");
lqraph = connectLayers(lqraph,"Encoder-Stage-3-ReLU-2","Decoder-Stage-2-
Concatenation/in2");
lgraph = connectLayers(lgraph,"Encoder-Stage-4-ReLU-2","Encoder-Stage-4-
MaxPool");
lgraph = connectLayers(lgraph,"Encoder-Stage-4-ReLU-2","Decoder-Stage-1-
Concatenation/in2");
lgraph = connectLayers(lgraph, "Decoder-Stage-1-UpConv", "Decoder-Stage-1-
Concatenation/in1");
lgraph = connectLayers(lgraph, "Decoder-Stage-2-UpConv", "Decoder-Stage-2-
Concatenation/in1");
lgraph = connectLayers(lgraph, "Decoder-Stage-3-UpConv", "Decoder-Stage-3-
Concatenation/in1");
lgraph = connectLayers(lgraph, "Decoder-Stage-4-UpConv", "Decoder-Stage-4-
Concatenation/in1");
% analyzeNetwork(lgraph) % comment in to check structure and errors
plot(lgraph) % comment in to show network structure
%% Specify training options
options = trainingOptions('adam', ...
    'MaxEpochs',150, ...
    'InitialLearnRate',5e-4, ...
    'L2Regularization',1e-4,...
    'LearnRateSchedule', 'piecewise', ...
    'LearnRateDropPeriod',5,...
    'LearnRateDropFactor',0.95, ...
    'Plots', 'training-progress', ...
    'Verbose',true, ...
    'VerboseFrequency',20,...
    'Shuffle', 'every-epoch',...%
    'ExecutionEnvironment', 'auto',...
    'MiniBatchSize',MiniBatchSize);
%% Train and save the network
close all
% Specify location to save the network
saved network directory =
'D:\John\2021\MATLAB biostudies\MATLAB 3D UNET\3 Saved models\';
% Train the network
modelDateTime = datestr(now, 'dd-mmm-yyyy-HHMM');
[net,info] = trainNetwork(Training_ds,lgraph,options);
% Timestamp and save the network after training
save([saved network directory,'MLN Biostudies ',modelDateTime,'.mat'],'net','opt
```

ions','info');



3-D Unet architecture schematic. The network uses an input layer for the reflectance data of 64x64x64x1 (x, y, z, channels). The three-class Unet architecture uses an encoder depth of 4 with 64 filters at the level of the first encoder. The network uses complete up-convolutional expansion to yield outputted probability maps that are identically sized to the input layer.

```
% MATLAB SCRIPT: TEST 3D UNET
% All code, image-data and screen-cast tutorial videos available for download at
% the Biostudies database under accession number S-BSST742
clear
close all
clc
% Specify path to bioformats library
addpath('D:\John\2021\MATLAB biostudies\MATLAB 3D UNET\4 TEST Unet\bfmatlab\')
% Specify directory containing test data
Test data directory =
'D:\John\2021\MATLAB biostudies\MATLAB 3D UNET\1 ImageData\TEST\';
% Specify channel number in test data that contains reflectance information
RL channel number = 2;
% COMMIT REFLECTANCE TEST DATA TO DIRECTORY RESCALED [0 1] IN MAT FORMAT
if ~exist('mat_test_data', 'dir')
   mkdir('mat test data');
end
for loop = 1:1
   counter = sprintf('%03d',loop) ;
   % find metadata describing file
   reader = bfGetReader([Test data directory,'TEST ',counter,'.tif']);
   omeMeta = reader.getMetadataStore();
   number_of_channels = omeMeta.getChannelCount(0);
   stackSizeZ = omeMeta.getPixelsSizeZ(0).getValue(); % number of Z slices
   % Load reflectance information
   for zplane = 1:stackSizeZ
       iPlane = reader.getIndex(zplane -1, RL channel number -1, 0) + 1; %
because zplanes and channels are numbered from zero
       channel zimage{zplane} = bfGetPlane(reader, iPlane);
   end
   IM_DATA = cat(3,channel_zimage{:}) ;
   IM DATA = double(1*mat2gray(IM DATA, [0 65535])) ; % rescale zero-one
   save([pwd,'/mat test data/','TEST DATA ',counter,'.mat'], 'IM DATA');
end
clear
88
% load the trained 3D Unet network
load('D:\John\2021\MATLAB biostudies\MATLAB 3D UNET\3 Saved models\MLN Biostudie
s 27-May-2021-1355.mat') ;
% load the rescaled reflectance test data
load([pwd,'/mat_test_data/','TEST_DATA_001.mat']);
%% Patch the reflectance data through the network
****
% Set the patch sizes to be passed to the net
patchSize = [256 256 64];
% Segment blockwise then reassemble in full
% Define image dimensions
```

```
[height, width, depth, nChannel] = size(IM DATA);
patch = zeros([patchSize, nChannel], 'like', IM DATA);
number of height patches =ceil(height/patchSize(1));
number_of_width_patches =ceil(width/patchSize(2));
number of depth patches =ceil(depth/patchSize(3));
% Pad image to have dimensions as multiples of patchSize
height_pad = number_of_height_patches*patchSize(1);
width_pad = number_of_width_patches*patchSize(2);
depth pad = number of depth patches*patchSize(3);
% Amount to pad different dimensions of image
padSize(1) = height pad - height;
padSize(2) = width pad - width;
padSize(3) = depth pad - depth;
% Pad the image by correct amounts
im_pad = padarray (IM_DATA, padSize, 0, 'post');
% Preallocate some matrices to catch the probability maps
out Uncertainty Scores = zeros([size(im pad,1), size(im pad,2), size(im pad,3)],
'double'); % needs to match OutputType in semanticseq below
out scores from network all classes=zeros([size(im pad,1), size(im pad,2),
size(im_pad,3), 3], 'double');
% Loop through blocks of 'patchSize'
for loop height = 1:number of height patches
    for loop_width = 1:number_of_width_patches
        for loop_depth = 1:number_of_depth_patches
            start height position=(loop height-1)*patchSize(1)+1;
            end height position=loop height*patchSize(1);
            start width position=(loop width-1)*patchSize(2)+1;
            end_width_position=loop_width*patchSize(2);
            start_depth_position=(loop_depth-1)*patchSize(3)+1;
            end_depth_position=loop_depth*patchSize(3);
            patch to deploy=...
im pad(start height position:end height position,start width position:end width
position,start depth position:end_depth_position,:);
            % deploy net
            [patch_seg, Scores, allScores] = semanticseg(patch to deploy, net,
'OutputType', 'double',...
            'ExecutionEnvironment', 'auto');
out Uncertainty Scores(start height position:end height position,start width pos
ition:end width position,start depth position:end depth position)...
            =Scores;
out scores from network all classes(start height position:end height position,st
art width position:end width position, start depth position:end depth position,:)
. . .
            =allScores;
```

end

```
end
% Script is general for n classes to this point
out Pmap cat1 = out scores from network all classes(:,:,:,1);
out Pmap cat2 = out scores from network all classes(:,:,:,2);
out Pmap cat3 = out scores from network all classes(:,:,:,3);
% Remove padding from probability maps
out Uncertainty Scores = out Uncertainty Scores(1:height, 1:width, 1:depth);
out Pmap cat1 = out Pmap cat1(1:height, 1:width, 1:depth);
out Pmap cat2 = out Pmap cat2(1:height, 1:width, 1:depth);
out Pmap cat3 = out Pmap cat3(1:height, 1:width, 1:depth);
%% visualise network outputs
figure(2)
clf(figure(2))
ax1 = subplot(2,3,1);
imshow(out Pmap cat1(:,:,60),[])
title('LF-nuclei')
ax2 = subplot(2,3,2);
imshow(out Pmap cat2(:,:,60),[])
title('LF-actin')
ax3 = subplot(2,3,3);
imshow(out Pmap cat3(:,:,60),[])
title('Background/other')
ax4 = subplot(2,3,5);
imshow(out Uncertainty Scores(:,:,60),[])
title('Uncertainty')
linkaxes([ax1 ax2 ax3 ax4], 'xy')
%% Map to 16-bit and save for loading into CellProfiler
uil6 PMAP cat1 xy = uint16(65535*mat2gray(out Pmap cat1, [0 1]));
uil6 PMAP cat2 xy = uint16(65535*mat2gray(out Pmap cat2, [0 1])) ;
ui16_PMAP_cat3_xy = uint16(65535*mat2gray(out_Pmap_cat3, [0 1])) ;
data to save = cat(4,uil6 PMAP cat1 xy,uil6 PMAP cat2 xy,uil6 PMAP cat3 xy);
% Map to 16-bit and save for loading into CellProfiler
for loop = 1:3
   CP channel number = num2str(loop +2) ;
   MultiChImgTile = data to save(:,:,:,loop) ;
   % spec a fiji description
   fiji descr = ['ImageJ=1.52p' newline ...
               'images=' num2str(size(MultiChImgTile,3)*...
                                size(MultiChImgTile,4)*...
                                size(MultiChImgTile,5)) newline...
               'channels=' num2str(size(MultiChImgTile,4)) newline...
               'slices=' num2str(size(MultiChImgTile,3)) newline...
               'frames=' num2str(size(MultiChImgTile,5)) newline...
               'hyperstack=true' newline...
               'mode=grayscale' newline...
               'loop=false' newline...
               'min=0.0' newline...
               'max=65535.0']; % bitdepth spec
   t = Tiff(['Ch ',CP channel number,' Im ','001','.tif'],'w');
```

end

```
tagstruct.ImageLength = size(MultiChImgTile,1);
    tagstruct.ImageWidth = size(MultiChImgTile,2);
    tagstruct.Photometric = Tiff.Photometric.MinIsBlack;
    tagstruct.BitsPerSample = 16;
    tagstruct.SamplesPerPixel = 1;
    tagstruct.Compression = Tiff.Compression.None; %% lzw is not compatible w
CP4 out-of-box
    tagstruct.PlanarConfiguration = Tiff.PlanarConfiguration.Chunky;
    tagstruct.SampleFormat = Tiff.SampleFormat.UInt;
    tagstruct.ImageDescription = fiji_descr;
        for frame = 1:size(MultiChImgTile,5)
            for slice = 1:size(MultiChImgTile,3)
                for channel = 1:size(MultiChImgTile,4)
                    t.setTag(tagstruct)
                    t.write(im2uint16(MultiChImgTile(:,:,slice,channel,frame)));
                    t.writeDirectory(); % saves a new page in the tiff file
                end
            end
        end
        t.close()
end
```

Windows 10: Running the label-free cell segmentation deep learning scripts on an NVIDIA GPU using Python 3.6 and Tensorflow-gpu 1.9.0

In brief, it is recommended to use these deep learning files with:

- Python 3.6
- Tensorflow-gpu 1.9.0
- Keras 2.1.5
- Numpy 1.18.1
- Scipy 1.4.1
- Java SE Development Kit 11.0
- Python-bioformats 1.5.2
- CUDA Toolkit 9.0 / cuDNN v7.6.4

Installation Steps:

1. Install Visual Studio Express Community 2017

https://visualstudio.microsoft.com/dev-essentials/#software

- This is necessary to enable the install of the CUDA toolkit.
- At the link above, join Visual Studio Development Essentials (free sign-up).
- Use the search tool to find, download and install Visual Studio Express Community 2017.
- Using the recommended configurations at every step of the installation works fine.

Hicrosoft Visual Studio Subscr	iptions				jw2020 Sig
Your Downloads	• £03 ·	₽ ij	· /2 • 4	· 🗜 • 🎾 ·	
Benefits Downloads Product Keys	Subscriptions Get Help	Marketplace			
All Visual Studio Com	munity				×
FILTER BY PRODUCT FAMILY	DOWNLOADS (5)				Sort by: Release date \checkmark
Visual Studio 2013 Update 5	Visual Studio Community	2019 (version 16.11)			
Visual Studio 2015 Update 3	🔍 No key required	1 Info	Release date: 16/Nov/2021	x64 V Multiple Lang V exe V	Download 🛓
Visual Studio 2017 (version 15.9) Visual Studio 2019 (version 16.11)	Visual Studio Community	2022 (version 17.0)			
Visual Studio 2022 (version 17.0)	🔦 No key required	1 Info	Release date: 16/Nov/2021	x64 V Multiple Lang V exe V	Download 🛓
	Visual Studio Community	r 2017 (version 15.9)			
	🔦 No key required	1 Info	Release date: 09/Nov/2021	x64 V Multiple Lang V EXE V	Download 🛓
	Visual Studio Community	2015 with Update 3			
	A No key required	1 Info	Release date: 27/Jun/2016	x64 V English V DVD V	Download 🛓
	Visual Studio Community	2013 with Update 5			
	🔍 No key required	1 Info	Release date: 13/Nov/2015	x86 V English V DVD V	Download 🛓

- Restarting your PC after installing Visual Studio is a probably a good idea.

2. Install CUDA Toolkit 9.0 and accompanying patches for Windows 10.

- Tensorflow-GPU 1.9.0 requires CUDA 9.0 - not whatever the latest version of the toolkit is. - This is available at the Nvidia website "CUDA Toolkit Archive" --> select CUDA Toolkit 9.0 https://developer.nvidia.com/cuda-toolkit-archive

🔲 🞯 NVIDIA. DEVELOPER

CUDA Toolkit Archive

Home

Previous releases of the CUDA Toolkit, GPU Computing SDK, documentation and developer drivers can be found using the links below. Please select the release you want from the list below, and be sure to check www.nvidia.com/drivers for more recent production drivers appropriate for your hardware configuration.

Download Latest CUDA Toolkit Learn More about CUDA Toolkit 11

Latest Release

CUDA Toolkit 11.5.1 (November 2021), Versioned Online Documentation

Archived Releases

CUDA Toolkit 11.5.0 (October 2021), Versioned Online Documentation CUDA Toolkit 11.4.3 (November 2021), Versioned Online Documentation CUDA Toolkit 11.4.2 (September 2021), Versioned Online Documentation CUDA Toolkit 11.4.1 (August 2021), Versioned Online Documentation CUDA Toolkit 11.4.0 (June 2021), Versioned Online Documentation CUDA Toolkit 11.3.1 (May 2021), Versioned Online Documentation CUDA Toolkit 11.3.0 (April 2021), Versioned Online Documentation CUDA Toolkit 11.2.2 (March 2021), Versioned Online Documentation CUDA Toolkit 11.2.1 (Feb 2021), Versioned Online Documentation CUDA Toolkit 11.2.0 (Dec 2020), Versioned Online Documentation CUDA Toolkit 11.1.1 (Oct 2020), Versioned Online Documentation CUDA Toolkit 11.1.0 (Sept 2020), Versioned Online Documentation CUDA Toolkit 11.0 Update1 (Aug 2020), Versioned Online Documentation CUDA Toolkit 11.0 (May 2020), Versioned Online Documentation CUDA Toolkit 10.2 (Nov 2019), Versioned Online Documentation CUDA Toolkit 10.1 update2 (Aug 2019), Versioned Online Documentation CUDA Toolkit 10.1 update1 (May 2019), Versioned Online Documentation CUDA Toolkit 10.1 (Feb 2019), Online Documentation CUDA Toolkit 10.0 (Sept 2018), Online Documentation CUDA Toolkit 9.2 (May 2018), Online Documentation CUDA Toolkit 9.1 (Dec 2017). Online Documentation CUDA Toolkit 9.0 (Sept 2017), Online Documentation IDA Toolkit 8 0 GA2 IEeh 2017L Online Docum

- Select the target platform as Windows, X86_64, version 10 and the installer type as exe(local)

41



Download Installers for Windows 10 x86_64

The base installer is available for download below. There are 4 patches available. These patches require the base installer to be installed first.

> Base Installer	Download (1.4 GB) 📥
Installation Instructions:	
 Double click cuda_9.0.176_win10.exe Follow on-screen prompts 	
> Patch 1 (Released Jan 25, 2018)	Download (54.1 MB) 📥
cuBLAS Patch Update: This update to CUDA 9.0 includes new GEMM kernels optimized for the Volta architecture and improved heuristics to select GEMM kernels for given input sizes.	
> Patch 2 (Released Mar 5, 2018)	Download (54.7 MB) 📥
cuBLAS Patch Update: This update to CUDA 9 includes GEMM heuristics improvements to selects the most optimized algorithms for input sizes commonly used in Deep Learning RNNs. The update also includes other bug- fixes and performance enhancements.	
> Patch 3 (Released Jun 7, 2018)	Download (82.3 MB) 📩
cuBLAS Patch Update: This update to cuBLAS addresses issues with Convolutional Seq2Seq and RNN inference performance.	
> Patch 4 (Released Aug 6, 2018)	Download (56.2 MB) 📥
cuBLAS Patch Update: This update to cuBLAS includes optimized implementations of GEMV operations for mixed precision input and output types and important fixes to address performance issues.	
The checksums for the installer and patches can be found in Installer Checksums. For further information, see the Installation Guide for Microsoft Windows and the CUI	DA Quick Start Guide.

- First install the "base installer" following the standard, "express configurations" options.

- Then install Patches 1-4 sequentially in order by just following the on-screen prompts after each download.

3. Install cuDNN v7.6.4 for CUDA 9.0

- To download and install cuDNN, first join the NVIDIA Developer Program – which is free. https://developer.nvidia.com/cudnn-download-survey

- Once signed in, proceed the cuDNN download page and click archived cuDNN releases: https://developer.nvidia.com/rdp/cudnn-archive

cuDNN Download

Home

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

a) Agree To the Terms of the cuDNN Software License Agreement
Note: Please refer to the Installation Guide for release prerequisites, including supported GPU architectures and compute capabilities, before downloading.
For more information, refer to the cuDNN Developer Guide, Installation Guide and Release Notes on the Deep Learning SDK Documentation web page.

Download cuDNN v8.3.1 (November 22nd, 2021), for CUDA 11.5
Download cuDNN v8.3.1 (November 22nd, 2021), for CUDA 10.2
Archived cuDNN Releases

Ethical AI
NVIDIA's platforms and application frameworks enable developers to build a wide array of AI applications. Consider potential algorithmic bias when choosing or creating the models being deployed. Work with the model's developer to ensure that it meets the requirements for the relevant industry and use case; that the necessary instruction and documentation are provided to understand error rates, confidence intervals, and results; and that the model is being used under the conditions and in the manner intended.

- Scroll down to the option to download cuDNN v7.6.4 for CUDA 9.0 and download the library for Windows 10.

Download cuDNN v7.6.4 (September	27, 2019], for CUDA 9.0
Library for Windows,	Mac, Linux, Ubuntu(x86_64 architecture)
cuDNN Library for Windows 7	
cuDNN Library for Windows 10	
cuDNN Library for Linux	
cuDNN Runtime Library for Ubuntu1	5.04 (Deb)
cuDNN Developer Library for Ubuntu	16.04 (Deb)
cuDNN Code Samples and User Guid	e for Ubuntu16.04 (Deb)
cuDNN Runtime Library for Ubuntul	4.04 (Deb)
cuDNN Developer Library for Ubuntu	14.04 (Deb)
cuDNN Code Samples and User Guid	e for Ubuntu14.04 (Deb)
Library for Red Hat (x	(86_64)
cuDNN Runtime Library for RedHat/0	lentos 7.3 (RPM)
cuDNN Developer Library for RedHat	/Centos 7.3 (RPM)
cuDNN Code Samples and User Guid	e for RedHat/Centos 7.3 (RPM)

- Unzip the downloaded cuDNN .zip file.

- Inside are three files which need to be copied to the correct folder subdirectories of your Window 10 installation of the CUDA 9.0 toolkit:

- These files are cudnn64_7.dll, cudnn.h and cudnn.lib:

1. cudnn64_7.dll

Copy the file from the unzipped cuDNN download folder at e.g., <ur><unzipped_download>/cudnn-9.0-windows10-x64-v7.6.2.24\cuda\bin\cudnn64_7.dll

and paste it into the NVIDIA GPU computing toolkit v9.0 subdirectory 'bin' located at e.g., C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\bin\

2. cudnn.h

Copy the file from the unzipped cuDNN download folder at e.g., <ur>unzipped_download>/cudnn-9.0-windows10-x64-v7.6.2.24\cuda\ include\cudnn.h

and paste it into the NVIDIA GPU computing toolkit v9.0 subdirectory 'include' located at e.g., C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\include\

3. cudnn.lib

Copy the file from the unzipped cuDNN download folder at e.g., <unzipped_download>/cudnn-9.0-windows10-x64-v7.6.2.24\cuda\lib\x64\cudnn.lib

and paste it into the NVIDIA GPU computing toolkit v9.0 subdirectory 'include' located at e.g., C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\lib\x64\



4. Check that the CUDA environment variables are set in Windows 10

- On the right-side, of the dialogue that opens, click "Advanced system settings"

Settings		- B X
û Home	About	
Find a setting P	Your PC is being monitored and	This page has a few new settings
System	protected. See details in Windows Security	Some settings from Control Panel have moved here, and you can copy your PC info so it's easier to share.
C Display	Device specifications	Related attrices
40 Sound		Related settings
Notifications & actions	OMEN by HP Desktop PC 880-p1xx Device name DESKTOP-1998L7D	Device Manager
∂ Focus assist	Processor Intel/R) Core(TM) (7-8700K CPU (8) 3 70GHz - 3.70 GHz	Remote desktop
() Power & sleep	Installed RAM 16.0 GB Device ID 7DB16A01-6579-4FCF-AD9A-3446341A2189	
🖙 Storage	Product ID 00329-10180-00000-AA484 System type 64-bit operating system, z64-based processor	Advanced system settings
ानु Tablet	Pen and touch No pen or touch input is available for this display	
H Multi-tasking	Copy	Get help
部 Projecting to this PC	Rename this PC	
X Shared experiences	Windows specifications	
1 Clipboard	Edition Windows 10 Enterprise Version 20H2	
> Remote Desktop	Installed on 09/11/2021 OS build 15042.1348	
③ About	Experience Windows Feature Experience Pack 120.2212.39200	
	Сору	
	Change the product key or upgrade your edition of Windows	
	Read the Microsoft Services Agreement that applies to our services	
	Read the Microsoft Software Licence Terms	

•					
Computer Name	Hardware	Advanced	System Protection	Remote	
You must be log	gged on as a	an Administrat	or to make most of th	ese changes	3.
Visual effects	. processor s	schedulina, m	emory usage and virt	ual memory	
				,	
				Settings	
User Profiles					
Desktop setti	ngs related to	o your sign-in			
			_		
				Settings	
Start-up and F	Recovery				
System start-u	up, system fa	ilure and deb	ugging information		
				-	
				Settings	
		i i	-		
			Environme	nt Variables.	

- In the "System Properties" dialogue box, click "Environment Variables" at the bottom

×

Environment Variables

ariable	Value			
P_JAVA_HOME	C:\Program Files (x86)\CellProfiler\java			
RE_HOME DneDrive DneDriveCommercial lath EMP MP	C:NPROGRA-1/Java/JRE18-1.0_2 C:\Users\Claire\OneDrive - University of Cambridge C:\Users\Claire\OneDrive - University of Cambridge C:\Users\Claire\AppData\Local\Programs\Python\Python36\Scripts\;C:\ C:\Users\Claire\AppData\Local\Temp C:\Users\Claire\AppData\Local\Temp	Jsers\Claire\Appl	Data\Local\Progra	ms\Python\Py
		New	5 12	
em variables ariable DMSnec UDA_PATH	Value C\WWNDOWS\system32\cmd.exe C\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0	IVEW	Edit	Delete
tem variables ariable COMSDEC UDA_PATH CUDA_PATH_V9_0 MIVED 243	Value C-\WINDOWS\sustem32\cmd.exe C-\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0 C-\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0 C-\WINGOWSYNETD2-UNVERSUMETSUMETOBIA	11EW	Edit	Delete
tem variables ariable iomSoec UDA_PATH UDA_PATH_V9_0 INVEDAT RE_HOME IUMEES_OF_PROCESSORS IVCUDASAMPLES_ROOT IVCUDASAMPLES_OOT IVCUDASAMPLES_OOT IVCUDASAMPLES_OOT INICOSSOR_ESO_ROOT INIT ROCESSOR_ARCHITECTURE ROCESSOR_ARCHITECTURE ROCESSOR_REVISION SModulePath	Value C-WMINDCWS\system32\cmd.exe C:\Program Files\WIDIA GPU Computing Toolkit\CUDA\v9.0 C:\Program Files\WIDIA GPU Computing Toolkit\CUDA\v9.0 C:\Program Files\WIDIA GPU Computing Toolkit\CUDA\v9.0 C:\Program Data\WIDIA Corporation\CUDA Samples\v9.0 C:\Program Data\WIDIA Corporation\CUDA Samples\v9.0 C:\Program Data\WIDIA Corporation\CUDA Samples\v9.0 C:\Program Files\WIDIA GPU Computing Toolkit\CUDA\v9.0\bin;C:\Prog .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC IM AMD64 Intel64 Family 6 Model 158 Stepping 10, GenuineIntel 6 9ela %ProgramFiles%\WindowsPowerShell\Modules;C\WINDOWS\system32	gram Files\NVIDIA	A GPU Computing	Toolkit\CU

- Make sure in the <u>bottom window</u> (labelled "System Variables") that variables named CUDA_PATH and CUDA PATH V9.0 exist and point to the correct locations e.g.,

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\bin

and

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\libnvvp

respectively.

<u>- These paths should be auto-installed</u>. If they are missing on your system, they can be added by clicking the "New" button below the system variables <u>bottom</u> panel and entering the name and path into the dialogue box for your system.

5. Install Python 3.6.8

- To install Python version 3.6.8 navigate to the previous release at: <u>https://www.python.org/downloads/release/python-368/</u>

	🥏 Py	thon Rele	ase Python 3.6.8 Py 🗙	+		-	D
\leftarrow	\rightarrow	С	https://www.pyt	hon.org/downloads/release/python-368/	€=	Ē	

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		48f393a04c2e66c77bfc114e589ec630	23010188	SIG
XZ compressed source tarball	Source release		51aac91bdf8be95ec0a62d174890821a	17212420	SIG
macOS 64-bit/32-bit installer	macOS	for Mac OS X 10.6 and later	eb1a23d762946329c2aa3448d256d421	33258809	SIG
macOS 64-bit installer	macOS	for OS X 10.9 and later	786c4d9183c754f58751d52f509bc971	27073838	SIG
Windows help file	Windows		0b04278f5bdb8ee85ae5ae66af0430b2	7868305	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	73df7cb2f1500ff36d7dbeeac3968711	7276004	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	72f37686b7ab240ef70fdb931bdf3cb5	31830944	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	39dde5f535c16d642e84fc7a69f43e05	1331744	SIG
Windows x86 embeddable zip file	Windows		60470b4cceba52094121d43cd3f6ce3a	6560373	SIG
Windows x86 executable installer	Windows		9c7b1ebdd3a8df0eebfda2f107f1742c	30807656	SIG
Windows x86 web-based installer	Windows		80de96338691698e10a935ecd0bdaacb	1296064	SIG

- Scroll to the bottom of the page and download the "Windows x86-64 executable installer" with description "AMD64/EM64T/x64".

- Once downloaded, follow the on-screen prompts to install Python 3.6.8.



- Check the box to add Python to the Windows path.

6. Install Java Development Kit 11

- This is used by Python-bioformats to enable read/write of image-data.

- Download and install Java Development kit 11

https://www.oracle.com/java/technologies/downloads/#java11

Java S You must accept the Oracle Tec Thank you for accepting the Java SE;	SE Develo chnology Netw download th Oracle Technol you may now	pment Kit 11.0.5 Fork License Agreement for Oracle Java SE to his software. logy Network License Agreement for Oracle download this software.
Product / File Description	File Size	Download
Linux	147.82 MB	jdk-11.0.5_linux-x64_bin.deb
Linux	154.47 MB	• jdk-11.0.5_linux-x64_bin.rpm
Linux	171.62 MB	jdk-11.0.5_linux-x64_bin.tar.gz
macOS	166.73 MB	jdk-11.0.5_osx-x64_bin.dmg
macOS	167.06 MB	jdk-11.0.5_osx-x64_bin.tar.gz
Solaris SPARC	188.32 MB	jdk-11.0.5_solaris-sparcv9_bin.tar.gz
Windows	151.39 MB	jdk-11.0.5_windows-x64_bin.exe
Windows	171.47 MB	jdk-11.0.5_windows-x64_bin.zip

- Once Java SE Development Kit 11 is installed, set the **JAVA_HOME** environment variable and add the Java development kit to the Windows path. To do this, as above, open a File Explorer window, right-click on the '**This PC**' option and select '**Properties**' from the drop-down menu.



The control panel will pop up as a separate window. Select '**Advanced system settings'** from the list appearing at the right of the window.

In the 'system properties' dialogue that opens, select '**Environment Variables**'. This will cause the environment variables window to appear. To set the 'JAVA_HOME' variable, Click the '**New**' button option at the bottom of this window in the '<u>System variables'</u> section.

Name the new variable 'JAVA_HOME' and use the '**Browse Directory**' option to specify the path to JDK 11. Select '**OK**' to create this new variable:

mputer Name Hardware Advanced System Protection Remote	User variables for Claire		
You must be logged on as an Administrator to make most of these changes.	Variable	Value	^
Performance	JRE HOME	C:\PROGRA~1\Java\JRE18~1.0 2	
Visual effects, processor scheduling, memory usage and virtual memory	OneDrive	C:\Users\Claire\OneDrive - University of Cambridge	
	OneDriveCommercial	C:\Users\Claire\OneDrive - University of Cambridge	
Settings	Path	C:\Users\Claire\AppData\Local\Programs\Python\Python36\Scripts	
Llass Profiles	TEMP	C:\Users\Claire\AppData\Local\Temp	
Dealstee estimos minted to vour sign in	TMP	C:\Users\Claire\AppData\Local\Temp	
Desktop settings related to your signan			~
Settings		New Edit Delete	
Start-up and Recovery	Surtem variabler		
System start-up, system failure and debugging information	System variables		
0	Variable	Value	^
Settings	ComSpec	C:\WINDOWS\system32\cmd.exe	
	CUDA_PATH	C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0	
Environment Variables	CUDA_PATH_V9_0	C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0	
	DriverData	C:\Windows\System32\Drivers\DriverData	
	JRE HOME	C:\PROGRA~1\Java\JRE18~1.0_2	
OK Cancel Apply	NUMBER_OF_PROCESSORS	12	
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C:\ProaramData\NVIDIA Corporation\CUDA Samples\v9.0	~
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C:\ProgramData\NVIDIA Corporation\CUDA Samples\v9.0	~
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C:\ProgramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete	•
OK Cancel Apply	NUMBER_OF_PROCESSORS	12 C:\ProgramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete	~
OK Cancel Apply	NUMBER_OF_PROCESSORS	12 C:\ProoramData\NVIDIA Corooration\CUDA Samples\v9.0 New Edit Delete	•
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C:\ProaramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete OK Cancel	•
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C:\ProaramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete OK Cancel	•
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C:\ProgramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete OK Cancel	•
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES_ROOT	12 C:\ProgramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete OK Cancel	•
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C:\ProoramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete OK Cancel	•
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT	12 C-\ProgramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete OK Cancel	•
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT New System Variable Variable name: JAVA_HO Variable value: C:\Progu	12 C:\ProaramData\NVIDIA Corooration\CUDA Samples\v9.0 New Edit Delete OK Cancel DME ram Files\Java\jdk-1.8.2	
OK Cancel Apply	NUMBER_OF_PROCESSORS NVCUDASAMPLES ROOT New System Variable Variable name: JAVA_HC Variable value: C:\Progr	12 C:\ProaramData\NVIDIA Corporation\CUDA Samples\v9.0 New Edit Delete OK Cancel	
To add JDK11 to the Windows path, Click on the exisiting 'Path' system variable and click the '**Edit**' button. A new window will appear. Click new and type '**%JAVA_HOME%\bin**'.

System Properties X	Environment Variables		× Edit environment variable	×
Computer Name Hardware Advanced System Protection Remote You must be logged on as an Administrator to make most of these changes.	User variables for Claire		C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\bin ^ C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\libnv	New
Performance	Variable	Value	C:\Program Files\Common Files\Oracle\Java\javapath	Edit
Visual effects processor scheduling memory usage and vidual memory	CP_JAVA_HOME	C:\Program Files (x86)\CellProfiler\java	C:\Program Files (x86)\Common Files\Oracle\Java\javapath	
ridda chocco, processor seriesaning, memory deage and rinda memory	JRE_HOME	C:\PROGRA~1\Java\JRE18~1.0_2	C:\windows\system32	Browse
Settinge	OneDrive	C:\Users\Claire\OneDrive - University of Cambridge	C:\windows	
com gen	OneDriveCommercial	C:\Users\Claire\OneDrive - University of Cambridge	C:\windows\System32\Wbem	Delete
User Profiles	Path	C:\Users\Claire\AppData\Local\Programs\Python\Python36\Scripts	C:\windows\System32\WindowsPowerShell\v1.0\	
Desktop settings related to your sign-in	TEMP	C:\Users\Claire\AppData\Local\Temp	C:\windows\System32\OpenSSH\	
	TMP	C:\Users\Claire\AppData\Local\Temp	C:\Program Files (x86)\Intel\Intel(R) Management Engine Compon	Move Up
Settings		New Edit Delete	C:\Program Files\Intel\Intel(R) Management Engine Components\	
		New Edit Delete	C:\Program Files (x86)\Intel\Intel(R) Management Engine Compon	Move Down
Start-up and Recovery			C:\Program Files\Intel\Intel(R) Management Engine Components\I	
System start-up, system failure and debugging information	System variables		C:\Program Files\MATLAB\R2021b\runtime\win64	
	Variable	Value	C:\Program Files\MATLAB\R2021b\bin	Edit text
Settings			C:\Program Files\MATLAB\R2020a\bin	
	NVCUDASAMPLES_ROUT	C:\ProgramData\NVIDIA Corporation\CUDA Samples\V9.0	C:\Program Files\Git\cmd	
Environment Variables	NVCUDASAMPLES9_0_KOOT	C:\ProgramData\NVIDIA Corporation\CUDA Samples\v9.0	I%JAVA HOME%\bin	
	OnlineServices	Unline Services	C:\WINDOWS\system32	
	OS .	Windows_INT	C:\WINDOWS	
OK Cased Asph	Path	C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\bin;C	C:\WINDOWS\System32\Wbem	
OK Candel Apply	PATHEXT	.COM; EXE; BAT; CMD; VBS; VBE; JS; JSE; WSF; WSF; MSC	C:\WINDOWS\System32\WindowsPowerShell\v1.0\	
	biattormcode	IM	C:\WINDOWS\System32\OpenSSH\	
		New Edit Delete	C:\Program Files\Microsoft SQL Server\120\Tools\Binn\	
			C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common	
a the light of		OK Cancel	ОК	Cancel

- Click '**OK**' to apply changes and close all opened windows.

7. Setup a virtual environment and install the dependencies necessary to run the code

- Download and unzip the "Python_biostudies" folder from the BioStudies project archive to a suitable location on your computer.

📙 🛃 🚽 = 20211118_Python_bic	ostudies				– 🗆 ×						
File Home Share View					~ (?)						
Pin to Quick Copy Paste access	ath nortcut	Move to Copy to Rename Organise	Properties	Select all Select none							
← → ↑ ⊡ Search 20211118_Python_biostudies ✓ O > Search 20211118											
Documents	* ^	Name		Date modified	а Туре						
CELLPROFILER_OUT	*	Python		29/11/2021 12	1:56 File folder						
TRAIN	*	requirements		29/11/2021 12	:58 Text Docum						
👩 Recycle Bin	*										
TRAIN_DATA	*										
20200429_TiO2FiguresBackup	*										
2_TRAIN_Unet											
2_TRAIN_Unet											
4_TEST_Unet											
4_TEST_Unet											
	~	<			>						
2 items 1 item selected											

- Run command prompt as administrator:

- To do this, Search "CMD", right-click on "Command Prompt" from the search results dialogue and click "Run as administrator":



- Using the 'cd' command, change the current directory to the unzipped python scripts folder downloaded from Biostudies e.g.,



- Make a new subdirectory 'venv' and initialise an instance of Python inside it e.g.,

C:\Users\John\20211118_Python_biostudies>python -m venv ./venv

- Activate the newly-created virtual environment e.g.,

C:\Users\John\20211118_Python_biostudies>venv\Scripts\activate.bat

- Update pip to the latest version (N.B., this is critical to the successful install of javabridge with Python 3.6)

(venv) C:\Users\John\20211118_Python_biostudies>python -m pip install --upgrade pip

- Using the "requirements.txt" file included in the BioStudies download, install the required dependencies and the Spyder 4.0 Integrated Development Environment (IDE) to the newly-created virtual environment e.g.,

(venv) C:\Users\John\20211118_Python_biostudies>pip install -r requirements.txt

8. Training a 2-D UNET model

- A screencast video is included with the BioStudies project archive.

- Download and unzip the Python BioStudies project archive at a suitable location on your computer.

📙 🛃 🚽 Python					-		×	
File Home Share View							~ 🕐	
Pin to Quick Copy Paste	Move Copy to * Copy	Move Copy to - to - copy Delete Rename New folder						
Clipboard	Organise	New	Open	Select				
\leftarrow \rightarrow \checkmark \uparrow \square \rightarrow This PC \rightarrow Windows	(C:) > Users > John > 20211118	Python_biostudies > Pyth	ion >	5 V	,⊂ Sea	rch Pytho	n	
	A Name	Date	modified Ty	pe	Size			
TRAIN	2D_UNET_PYTHON	18/11	/2021 13:43 Fil	le folder				
A Recycle Bin	B 3D_UNET_PYTHON	18/11	/2021 13:44 Fi	le folder				
TRAIN_DATA	*							
20200429_TiO2FiguresBackup	*							
2_TRAIN_Unet								
2_TRAIN_Unet								
4_TEST_Unet								
4_TEST_Unet								
OneDrive - University of Cambridge								
University of Cambridge								
2 items	v							

- Inside the 2D_UNET_Python folder, there is a sequentially-organised workflow starting with raw data and ending with a CellProfiler pipeline that obtains cell features from the label-free cell segmentation:

I Image: Imag					-	□ × ^ (?)
Ar Pin to Quick Copy Copy	Move Copy to Copy to Copy	New item • To be a sy access • Folder	Properties • Open • Edit • History Open	Select all Select none Invert selection	on	
\leftarrow \rightarrow \checkmark \uparrow \square \ll Users \Rightarrow John \Rightarrow 202	11118_Python_biostudies > Python	> 2D_UNET_PYTHON >	2D_PeyersPatch >	5 v	,⊃ Search	2D_Peyers
4_TEST_Unet	^ Name	Date	modified Ty	pe	Size	
OneDrive - University of Cambridge	1_ImageData	18/11	/2021 13:43 Fil	e folder		
🙀 University of Cambridge	2_TRAIN_Unet 3_Saved_models	07/12 29/11	/2021 21:08 Fil /2021 15:50 Fil	e folder e folder		
💻 This PC	4_TEST_Unet	29/11	/2021 16:00 Fil	e folder		
3D Objects	5_CellProfiler_cell_measures	urements 18/11	/2021 13:44 Fil	e folder		
Desktop						
🗄 Documents						
👆 Downloads						
b Music						
Pictures						
📑 Videos						
L Windows (C:)						
5 items	¥					

- In command prompt with the virtual environment active, type 'Spyder' to start the Spyder4 IDE.



- To train a 2-D UNET model, set the Spyder working directory to the "2_TRAIN_Unet" directory. Open the "A_TRAIN_UNET" script.



Update the path saving the training data and pixel classification labels in numpy format.
This should be inside the "2_TRAIN_Unet" directory.



- The script will save the model along with checkpoint values taken during training. You should specify where these files are to be saved. This should be folder 3 of the workflow: ("3_Saved_models").



- To start model training, click the green arrow button at the top of the Spyder dialogue or type the script name at the command line.



- Model training takes several hours (~ 4h on a NVIDIA GTX 1080 Ti GPU)

- The newly-trained model will be saved in the "3_Saved_models" directory.

9. Testing a pretrained 2-D model using unseen data

- A screencast video is included with the BioStudies project archive.

- Change the Spyder working directory to the "4_Test_Unet" directory. Open the

"A_TEST_UNET" script.

Spyder (Pyth	non 3.6)								
File Edit Search	n Source Run De	ebug Consoles P	rojects Tools View	v Help					
🗅 🗁 🖪		🧿 🕨 🖬	🔁 💵 😋			S 🔎	-	€ →	C:\Users\John\20211118_Python_biostudies\Python\20_UNET_PYTHON\20_PeyersPatch\4_TEST_Unet

- Change the path to the unseen test image (located in the "1_ImageData" folder).



- Specify which pretrained network to use (pretrained networks are located in the "3_Saved_models" folder

pretrained_model = unet()
pretrained_model.load_weight;('C:/Users/John/20211118_Python_biostudies/Python/2D_UNET_PYTHON/2D_PeyersPatch/3_Saved_models/my_h5_model_fully_trained.h5'

- Clicking the green arrow button or type the file name at the command line to process the unseen reflectance data with the selected pretrained network:



- Probability maps for the LF-nuclei, LF-actin and Background/Other classes will be saved in the working directory named ready for input into the CellProfiler pipeline (filenames Ch_5_lm_001, Ch_6_lm_001, Ch_7_lm_001, respectively).



10. Training a 3-D UNET Model

- A screencast video is included with the BioStudies project archive.

- Inside the 3D_UNET_Python folder, there is a sequentially-organised workflow starting with raw data and ending with a CellProfiler pipeline that obtains 3-D cell features from the label-free cell segmentation:

📙 🛃 📮 🗧 3D_UNET_PYTHON				- 🗆 ×
File Home Share View				^ (
★ Cut ★ Copy Paste Copy path Copy Paste shortcut Clipboard	Move Copy to ~ Copy to ~ Copy organise	New item • Easy access • New Open	n Select all Select none Invert selection Select	1
← → ~ ↑ 🛄 « John > 20211118_Pyt	hon_biostudies > Python > 3D_UNET_PYT	HON > 3D_UNET_PYTHON >	ٽ ~	
4_TEST_Unet	^ Name	Date modified	Туре	Size
OneDrive - University of Cambridge	1_ImageData	18/11/2021 13:44	File folder	
R. University of Cambridge	2_TRAIN_Unet	30/11/2021 13:46	File folder	
in University of Cambridge	3_Saved_models	07/12/2021 17:50	File folder	
This PC	4_TEST_Unet	07/12/2021 17:39	File folder	
🗊 3D Objects	5_CellProfiler_cell_measurements	18/11/2021 13:45	File folder	
E. Desktop				
Documents				
Downloads				
b Music				
Pictures				
Videos				
Windows (C:)				
5 items	v			

56

- To train a 3-D UNET model, open Spyder and set the working directory to the "2_TRAIN_Unet" directory. Open the "A_TRAIN_UNET_3D" script.



- Specify the checkpoint path during model training:



- Specify the location to save the model when training completes. This should be the "3_Saved_models" directory.



- Click "Run" to commence model training.
- This takes considerable time (approximately 6h on a NVIDIA GTX 1080 Ti GPU card).



11. Testing a pretrained 3-D UNET Model using unseen data

- A screencast video is included with the BioStudies project archive.

- Change the Python working directory to the "4_Test_Unet" directory. Open the

"A_TEST_UNET_3D" script.



- Specify the location of the unseen test image-data. This is located inside the "1_ImageData" folder at "1_ImageData/TEST/".

- Specify the channel containing the reflectance information (here, '2').

- Specify where the unseen data should be stored upon conversion to Numpy format. This should be inside the "4_Test_Unet" folder.

	#%% Specify directory containing test data			
58	Test_data_directory = 'C:/Users/John/20211118_Python_biostudies/Python/3D_UNET_	PYTHON/3D	UNET_PYTHON/1	_ImageData/TEST/'
	# Specify channel number in test data that contains reflectance information			
	RL channel number = 2;			
	#COMMIT REFLECTANCE TEST DATA TO DIRECTORY RESCALED [0 1] IN MAT FORMAT			
	MYDIR ='C:/Users/John/20211118 Python biostudies/Python/3D UNET PYTHON/3D UNET	PYTHON/4 1	EST Unet/npy	test data/'
	CHECK_FOLDER = os.path.isdir(MYDIR)			
66	Contraction and the second s			

- Specify a pretrained 3-D Unet model from the "3_Saved_models" directory.

285 206 #XX load the trained 3D Unet network 207 #model.load_weights("C:/Users/John/20211118_Python_biostudies/Python/3D_UNET_PYTHON/3D_UNET_PYTHON/3_Saved_models/my_h53D_model.h5") 208 pretrained_model.load_weights("C:/Users/John/20211118_Python_biostudies/Python/3D_UNET_PYTHON/3D_UNET_PYTHON/3_Saved_models/my_h53D_model_updated.h5") 209 pretrained_model.load_weights("C:/Users/John/20211118_Python_biostudies/Python/3D_UNET_PYTHON/3D_UNET_PYTHON/3_Saved_models/my_h53D_model_updated.h5")

- Click the green arrow to process the unseen reflectance data with the 3-D Unet model.

۲	Spyder	(Pytho	on 3.6)																
File	Edit S	earch	Sourc	e Run	Debug	g Cons	soles	Project	s Tool	s View	Help								
Ľ			8		@			2	₽	Ç	M	۲:	G	c≡	₩		K 2	æ	è

- Probability maps for the LF-nuclei, LF-actin and Background/Other classes will be saved as multipage .TIFF files in the working directory named ready for input into the CellProfiler pipeline (filenames Ch_3_lm_001, Ch_4_lm_001, Ch_5_lm_001, respectively).

I Image: Imag	anage 4_TEST_Unet				-	□ × ^ (2)
Pin to Quick access Copy Paste Cipiboard	Move Copy to* to* Organise	New item •	Properties Open	Select all Select nor Invert sele	ection	
← → ~ ↑ 📙 « 20211118_Python_biost	tudies > Python > 3D_UNET_PY	THON > 3D_UNET_PYTHO	N > 4_TEST_Unet >	5 v	, Searc	ch 4_TEST_Unet
4_TEST_Unet	^ Name ^	Date	Туре	Size	Tags	
OneDrive - University of Cambridge	npy_test_data	18/11/2021 13:45	File folder			
University of Cambridge	A_TEST_UNET_3D	07/12/2021 17:39	Python File TIFF File	16 KB 219 173 KB		
💻 This PC	Ch_4_Im_001	07/12/2021 15:26	TIFF File	219,173 KB		
🇊 3D Objects	Ch_5_Im_001	07/12/2021 15:26	TIFF File	219,173 KB		
Desktop	Predictions	07/12/2021 15:19	PNG File	290 KB		
Documents						
Downloads Music						
E Pictures						
Videos						
Kindows (C:)	~					
6 items 1 item selected 214 MB						

58

```
% PYTHON SCRIPT: TRAIN 2D UNET
% All code, image-data and screen-cast tutorial videos available for download at
% the Biostudies database under accession number S-BSST742
# -*- coding: utf-8 -*-
Created on Mon Sep 20 23:58:04 2021
@author: Paul
#Import all required modules
from PIL import Image
import cv2
import matplotlib.pyplot as plt
import matplotlib.pyplot as plt
import skimage
from skimage import data
from skimage.filters import threshold otsu
import cv2
import os
import numpy as np
import skimage.transform as trans
import tensorflow
from tensorflow.keras.models import *
from tensorflow.keras.layers import *
from tensorflow.keras.optimizers import *
from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler
from tensorflow.keras import backend as keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import sklearn.feature extraction
from sklearn.datasets import load sample image
from sklearn.feature extraction import image
from datetime import datetime
from keras.utils import to categorical
from tensorflow.python.keras.optimizers import *
import skimage.io
import h5py # this was missing
#88
# Read in the reflectance data for training
Training data holder =
skimage.io.imread('C:/Users/John/20211118 Python biostudies/Python/2D UNET PYTHO
N/2D PeyersPatch/1 ImageData/TRAIN IMAGE.tif')
Training data = Training data holder[0,0,:,:,2]
#Read in the mask to isolated just the lymphoid tissue
mask = Training data holder[0,0,:,:,3]
thresh = threshold otsu(Training data)
mask = mask > thresh
mask = mask*1
mask = np.uint16(mask)
# Apply the mask to the training data
```

```
Training data = np.multiply(Training data,mask);
Training data = np.double(Training data);
# Rescale the training data in the interval [0 1]
Training data norm holder = np.zeros((8551, 5701), np.double)
Training data norm = cv2.normalize(Training data, Training data norm holder ,
1.0, 0.0, cv2.NORM MINMAX)
#Read in the pixel-class labels created from the nuclei and actin staining
Training labels holder =
skimage.io.imread('C:/Users/John/20211118 Python biostudies/Python/2D UNET PYTHO
N/2D PeyersPatch/1_ImageData/TRAIN_LABELS.png') ;
Training labels = np.where(Training labels holder == 3, 0,
Training labels holder)
#% Visually inspect the data and labels
fig1, (ax1, ax2) = plt.subplots(1, 2) # figure1
ax1.imshow(Training data norm)
ax1.set title('Reflectance data')
ax2.imshow(Training labels)
ax2.set title('Pixel classification labels')
#%% Once happy with labels and data, commit the reflectance information data to
sub-directory in .npy format
MYDIR =
'C:/Users/John/20211118 Python biostudies/Python/2D UNET PYTHON/2D PeyersPatch/2
_TRAIN_Unet/npy_training_data/
CHECK FOLDER = os.path.isdir(MYDIR)
# If folder doesn't exist, then create it.
if not CHECK_FOLDER:
   os.makedirs(MYDIR)
np.save(os.path.join(MYDIR, 'Training_data'), Training_data_norm)
# Create matrices to store training data
Patching data = np.zeros((8551, 5701,2))
Patching data[:,:,0] = Training data norm
Patching_data[:,:,1] = Training_labels
#Create a training data comprising matching patches (256x256 pixels) of image-
data and training labels
Patched images =
sklearn.feature extraction.image.extract patches 2d(Patching data, patch size =
[256, 256], max patches=12000, random state=None)
#%%
# CREATE THE UNET
# input layer 256x256x1
# encoder depth 4
# 64 filters at the level of the first encoder
def unet(pretrained weights = None,input size=(256,256,1), n class=3):
```

```
inputs = tensorflow.keras.Input(shape=input size)
    conv1 = Conv2D(64, 3, activation = 'relu', dilation rate=2,padding = 'same',
kernel initializer = 'he normal')(inputs)
    conv1 = BatchNormalization()(conv1)
    conv1 = Conv2D(64, 3, activation = 'relu', dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(conv1)
    conv1 = BatchNormalization()(conv1)
    pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)
    conv2 = Conv2D(128, 3, activation = 'relu', dilation rate=2,padding =
'same', kernel initializer = 'he normal')(pool1)
   conv2 = BatchNormalization()(conv2)
    conv2 = Conv2D(128, 3, activation = 'relu', dilation rate=2, padding =
'same', kernel_initializer = 'he_normal')(conv2)
    conv2 = BatchNormalization()(conv2)
    pool2 = MaxPooling2D(pool size=(2, 2))(conv2)
    conv3 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(pool2)
   conv3 = BatchNormalization()(conv3)
    conv3 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv3)
    conv3 = BatchNormalization()(conv3)
    pool3 = MaxPooling2D(pool size=(2, 2))(conv3)
    conv4 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(pool3)
    conv4 = BatchNormalization()(conv4)
    conv4 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv4)
    conv4 = BatchNormalization()(conv4)
    drop4 = Dropout(0.5)(conv4, training=True)
    pool4 = MaxPooling2D(pool size=(2, 2))(drop4)
    conv5 = Conv2D(1024, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(pool4)
    conv5 = BatchNormalization()(conv5)
    conv5 = Conv2D(1024, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv5)
    conv5 = BatchNormalization()(conv5)
    drop5 = Dropout(0.5)(conv5, training=True)
    up6 = Conv2D(512, 2, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(UpSampling2D(size = (2,2))(drop5))
   merge6 = concatenate([drop4,up6], axis = 3)
    conv6 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(merge6)
    conv6 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv6)
    up7 = Conv2D(256, 2, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(UpSampling2D(size = (2,2))(conv6))
   merge7 = concatenate([conv3,up7], axis = 3)
    conv7 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(merge7)
    conv7 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(conv7)
    up8 = Conv2D(128, 2, activation = 'relu', padding = 'same',
```

kernel_initializer = 'he_normal')(UpSampling2D(size = (2,2))(conv7))

```
merge8 = concatenate([conv2,up8], axis = 3)
   conv8 = Conv2D(128, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he_normal')(merge8)
    conv8 = Conv2D(128, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv8)
    up9 = Conv2D(64, 2, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(UpSampling2D(size = (2,2))(conv8))
   merge9 = concatenate([conv1,up9], axis = 3)
    conv9 = Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(merge9)
    conv9 = Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv9)
   conv10 = Conv2D(n class, 1, activation = 'softmax')(conv9)
   model = tensorflow.keras.Model(inputs = inputs, outputs = conv10)
   model.compile(optimizer = Adam(lr = 0.0001),loss =
'sparse categorical crossentropy', metrics = ['sparse categorical accuracy'])
    if(pretrained weights):
     model=keras.models.load model(pretrained weights)
    return model
#%% Split data into image and label data and reshape ready for training
trainX = Patched images[:,:,:,0]
trainX = np.asarray(trainX).reshape((12000, 256, 256,1))
trainY = Patched images[:,:,:,1]
trainY = np.asarray(trainY).reshape((12000, 256, 256, 1))
#Define model for data
model = unet(pretrained weights = None,input size = (256,256,1),n class=3)
# Define augmentation options
def get train augmented(trainX=trainX, trainY=trainY, BATCH SIZE=12):
    aug_X = ImageDataGenerator(rotation_range=360, zoom_range=[1,1],
width_shift_range=[0,0], height_shift_range=[0,0], horizontal_flip=True,
vertical_flip = True, shear_range = 0,fill_mode = "constant",cval=0.0)
    aug Y = ImageDataGenerator(rotation range=360, zoom range=[1,1],
width shift range=[0,0], height_shift_range=[0,0], horizontal_flip=True,
vertical_flip = True, shear_range = 0,fill_mode = "constant",cval=0.0)
    aug X.fit(trainX, augment=True, seed=1)
    aug_Y.fit(trainY, augment=True, seed=1)
    X train augmented = aug X.flow(trainX, batch size=BATCH SIZE, shuffle=True,
seed=1)
```

```
Y_train_augmented = aug_Y.flow(trainY, batch_size=BATCH_SIZE, shuffle=True,
seed=1)
    train generator = zip(X \text{ train augmented}, Y \text{ train augmented})
    for (X train augmented, Y train augmented) in train generator:
        yield (X train augmented, Y train augmented)
#%% Define path to safe network, time stamp network
checkpoint filepath =
'C:/Users/John/20211118 Python biostudies/Python/2D UNET PYTHON/2D PeyersPatch/3
Saved models/'
now = datetime.now()
dt string = now.strftime("%d/%m/%Y%H:%M:%S")
NAME =dt string
Name_file = 'PeyersPatchBiostudies '
checkpoint filepath name = checkpoint filepath + Name file+dt string
checkpoint dir = os.path.dirname(checkpoint filepath name)
#Specify that weights are to be saved
model checkpoint callback = tensorflow.keras.callbacks.ModelCheckpoint(
    filepath=checkpoint dir,
    save weights only=True,
    monitor='val accuracy',
    mode='max',
    save best only=True)
# Specify training options
BATCH SIZE = 12
NUM EPOCHS = 50
#Augment training data
train generator = get train augmented(trainX=trainX, trainY=trainY,
BATCH SIZE=BATCH SIZE)
#Train and save the network
history = model.fit generator(train generator,
steps per epoch=len(trainX)/(BATCH SIZE*2), epochs=NUM EPOCHS,
callbacks=[model checkpoint callback])
#Plot progress # this plots over the view of the data
plt.figure(2)
plt.plot(history.history['sparse categorical accuracy'])
plt.title('Model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.savefig('Training plot.png')
plt.show()
#Save final fully trained model as a .h5 file
model.save("C:/Users/John/20211118 Python biostudies/Python/2D UNET PYTHON/2D Pe
yersPatch/3 Saved models/my h5 model.h5")
checkpoint filepath name h5 = checkpoint filepath + Name file+dt string+'.h5'
model.save(checkpoint filepath name h5)
```

```
% PYTHON SCRIPT: TEST 2D UNET
% All code, image-data and screen-cast tutorial videos available for download at
% the Biostudies database under accession number S-BSST742
# -*- coding: utf-8 -*-
Created on Tue Sep 21 02:46:52 2021
@author: Paul
#Import all required modules
import skimage
from skimage import data
from skimage.filters import threshold otsu
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import sklearn.feature extraction
from sklearn.datasets import load sample image
from sklearn.feature_extraction import image
import matplotlib.cm as cm
import matplotlib.pyplot as plt
import matplotlib.cbook as cbook
from matplotlib.path import Path
from matplotlib.patches import PathPatch
import PIL
import numpy as np
import cv2
import tensorflow
from tensorflow.keras.models import *
from tensorflow.keras.layers import *
from tensorflow.keras.optimizers import *
from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler
from tensorflow.keras import backend as keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import sklearn.feature extraction
from sklearn.datasets import load sample image
from sklearn.feature extraction import image
from datetime import datetime
from keras.utils import to categorical
from tensorflow.python.keras.optimizers import *
import skimage.io
#%% Load the reflectance data from the unseen test image
TEST DATA =
skimage.io.imread('C:/Users/John/20211118 Python biostudies/Python/2D UNET PYTHO
N/2D PeyersPatch/1 ImageData/TEST IMAGE.tif');
Test_data = TEST_DATA[0,0,:,:,2]
# Load the mask for the lymphoid tissue region
Mask = TEST DATA[0,0,:,:,3]
thresh = threshold otsu(Test data)
Mask = Mask > thresh
Mask = Mask*1
Mask = np.uint16(Mask)
# Rescale the test data [0 1]
# Mask the rescaled data
Test data = np.multiply(Test data,Mask);
Test data = np.double(Test data);
```

```
Test_data_norm_holder = np.zeros((11247, 7610),np.double)
Test data rescaled = cv2.normalize(Test data, Test data norm holder , 1.0, 0.0,
cv2.NORM MINMAX)
#88
# Define model for testing
def unet(pretrained weights = None, input size=(256,256,1), n class=3):
   inputs = tensorflow.keras.Input(shape=input size)
   conv1 = Conv2D(64, 3, activation = 'relu', dilation_rate=2,padding = 'same',
kernel initializer = 'he_normal')(inputs)
   conv1 = BatchNormalization()(conv1)
   conv1 = Conv2D(64, 3, activation = 'relu', dilation rate=2,padding = 'same',
kernel initializer = 'he normal')(conv1)
   conv1 = BatchNormalization()(conv1)
   pool1 = MaxPooling2D(pool size=(2, 2))(conv1)
   conv2 = Conv2D(128, 3, activation = 'relu', dilation rate=2, padding =
'same', kernel_initializer = 'he_normal')(pool1)
   conv2 = BatchNormalization()(conv2)
   conv2 = Conv2D(128, 3, activation = 'relu', dilation_rate=2, padding =
'same', kernel initializer = 'he normal')(conv2)
   conv2 = BatchNormalization()(conv2)
   pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)
   conv3 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(pool2)
   conv3 = BatchNormalization()(conv3)
   conv3 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv3)
   conv3 = BatchNormalization()(conv3)
   pool3 = MaxPooling2D(pool_size=(2, 2))(conv3)
   conv4 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(pool3)
   conv4 = BatchNormalization()(conv4)
   conv4 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(conv4)
   conv4 = BatchNormalization()(conv4)
   drop4 = Dropout(0.5)(conv4, training=True)
   pool4 = MaxPooling2D(pool size=(2, 2))(drop4)
   conv5 = Conv2D(1024, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(pool4)
   conv5 = BatchNormalization()(conv5)
   conv5 = Conv2D(1024, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv5)
   conv5 = BatchNormalization()(conv5)
   drop5 = Dropout(0.5)(conv5, training=True)
   up6 = Conv2D(512, 2, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(UpSampling2D(size = (2,2))(drop5))
   merge6 = concatenate([drop4,up6], axis = 3)
   conv6 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(merge6)
   conv6 = Conv2D(512, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv6)
```

```
up7 = Conv2D(256, 2, activation = 'relu', padding = 'same',
kernel initializer = 'he_normal')(UpSampling2D(size = (2,2))(conv6))
   merge7 = concatenate([conv3,up7], axis = 3)
   conv7 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(merge7)
   conv7 = Conv2D(256, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(conv7)
   up8 = Conv2D(128, 2, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(UpSampling2D(size = (2,2))(conv7))
   merge8 = concatenate([conv2,up8], axis = 3)
   conv8 = Conv2D(128, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(merge8)
   conv8 = Conv2D(128, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(conv8)
   up9 = Conv2D(64, 2, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(UpSampling2D(size = (2,2))(conv8))
   merge9 = concatenate([conv1,up9], axis = 3)
   conv9 = Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel initializer = 'he normal')(merge9)
   conv9 = Conv2D(64, 3, activation = 'relu', padding = 'same',
kernel_initializer = 'he_normal')(conv9)
   conv10 = Conv2D(n_class, 1, activation = 'softmax')(conv9)
   model = tensorflow.keras.Model(inputs = inputs, outputs = conv10)
   model.compile(optimizer = Adam(lr = 0.0001),loss =
'sparse_categorical_crossentropy', metrics = ['sparse_categorical_accuracy'])
   if(pretrained weights):
     model=keras.models.load model(pretrained weights)
   return model
#%%
# Patch the reflectance data through the network
# Set the patch sizes to be passed to the net
patchSize = [256, 256]
Test data rescaled =
np.asarray(Test data rescaled).reshape((Test data rescaled.shape[0],
Test data rescaled.shape[1],1))
# Segment blockwise then reassemble in full
# Define image dimensions
[height,width,nChannel] = np.shape(Test data rescaled)
```

```
patch = np.zeros([patchSize[0], patchSize[1],
nChannel],dtype=Test data rescaled.dtype);
# Pad image to have dimensions as multiples of patchSize
padSize = np.empty([1, 2])
padSize[0,0] = patchSize[0] - np.remainder(height, patchSize[0]);
padSize[0,1] = patchSize[1] - np.remainder(width, patchSize[1]);
Starting_shape = np.shape(Test_data_rescaled)
a1 = Starting_shape[0]+padSize[0,0]
a2 = Starting shape[1]+padSize[0,1]
im_pad = np.zeros([int(a1),int(a2)])
im pad[:Test data rescaled.shape[0],:Test data rescaled.shape[1]] =
Test_data_rescaled[:,:,0]
im pad = np.asarray(im pad).reshape((im pad.shape[0], im pad.shape[1],1))
[height_pad, width_pad, nChannel_pad] = np.shape(im_pad);
# Preallocate some matrices to receive the network outputs
out Uncertainty Scores = np.zeros([np.shape(im pad)[0], np.shape(im pad)[1]],
'double');
out_Pmap_cat1 = np.zeros([np.shape(im_pad)[0], np.shape(im_pad)[1]], 'double');
out Pmap cat2 = np.zeros([np.shape(im pad)[0], np.shape(im pad)[1]],
                                                                      'double');
out Pmap cat3 = np.zeros([np.shape(im_pad)[0], np.shape(im_pad)[1]], 'double');
# Load a pretrained network
pretrained model = unet()
pretrained model.load weights('C:/Users/John/20211118 Python biostudies/Python/2
D_UNET_PYTHON/2D_PeyersPatch/3_Saved_models/my_h5_model_fully_trained.h5')
# Loop through blocks of 'patchSize'
for loop in range(1, int(height_pad),int(patchSize[0])):
    print(loop)
    for j in range(1, int(width pad), int(patchSize[1])):
        for p in range(1,(nChannel+1)):
            PP3 = np.empty([1,256,256,1])
            im pad touse =im pad[(loop-1):((loop-1)+patchSize[0]),(j-1):((j-
1)+(patchSize[1]))]
            patch[:,:,0] = np.squeeze(im_pad_touse,axis=None)
            PP3[0,:,:,:]=patch
            # deploy net
            predictions = pretrained model.predict(PP3)
            scores = np.max(predictions,axis=3)
            predictions.argmax(axis=3)
            out Uncertainty Scores ((loop-1): (loop+patchSize[0]-1), (j-
1):(j+patchSize[1]-1)] = scores;
            out Pmap cat1[(loop-1):(loop+patchSize[0]-1), (j-1):(j+patchSize[1]-
1)] = predictions[0,:,:,0];
            out_Pmap_cat2[(loop-1):(loop+patchSize[0]-1), (j-1):(j+patchSize[1]-
1)] = predictions[0,:,:,1];
            out_Pmap_cat3[(loop-1):(loop+patchSize[0]-1), (j-1):(j+patchSize[1]-
1)] = predictions[0,:,:,2];
```

```
out Uncertainty Scores = out Uncertainty Scores[0:height, 0:width];
out Pmap cat1 = out_Pmap_cat1[0:height, 0:width];
out Pmap cat2 = out_Pmap_cat2[0:height, 0:width];
out Pmap cat3 = out Pmap cat3[0:height, 0:width];
image_to_plot = np.zeros((11247, 7610,3))
image to plot[:,:,0] = out Pmap cat1
image to plot[:,:,1] = out Pmap cat2
image_to_plot[:,:,2] = out_Pmap_cat3
#%% visualise network outputs
fig, (ax1, ax2, ax3, ax4) = plt.subplots(1, 4, sharey=True)
fig.set size inches(14, 12)
im = ax1.imshow(out Pmap cat1)
ax1.title.set text('Background/Other')
im = ax2.imshow(out Pmap cat2)
ax2.title.set text('LF-Actin')
im = ax3.imshow(out Pmap cat3)
ax3.title.set text('LF-Nuclei')
im = ax4.imshow(out_Uncertainty_Scores)
ax4.title.set_text('Uncertainty')
plt.savefig('Predictions.png')
#%% Map to 16-bit and save for loading into CellProfiler
# Background/other
data_norm_holder = np.zeros((11247, 7610),np.uint16)
uil6 PMAP cat1 xy = cv2.normalize(out Pmap cat1, data norm holder , 65535.0,
0.0, cv2.NORM MINMAX)
img_data2 = (ui16_PMAP_cat1_xy).astype(dtype=np.uint16)
rawtiff=PIL.Image.fromarray(img_data2)
rawtiff.save('Ch_7_Im_001'+'.tiff')
# LF-Actin
data norm holder = np.zeros((11247, 7610), np.uint16)
uil6 PMAP cat2 xy = cv2.normalize(out Pmap cat2, data norm holder , 65535.0,
0.0, cv2.NORM MINMAX)
img_data2 = (ui16_PMAP_cat2_xy).astype(dtype=np.uint16)
rawtiff=PIL.Image.fromarray(img_data2)
rawtiff.save('Ch 6 Im 001'+'.tiff')
# LF-Nuclei
data norm holder = np.zeros((11247, 7610),np.uint16)
uil6_PMAP_cat3_xy = cv2.normalize(out_Pmap_cat3, data_norm_holder , 65535.0,
0.0, cv2.NORM_MINMAX)
img data2 = (ui16 PMAP cat3 xy).astype(dtype=np.uint16)
rawtiff=PIL.Image.fromarray(img data2)
rawtiff.save('Ch 5 Im 001'+'.tiff')
```

% PYTHON SCRIPT: TRAIN 3D UNET % All code, image-data and screen-cast tutorial videos available for download at % the Biostudies database under accession number S-BSST742 # -*- coding: utf-8 -*-Created on Mon Jun 14 15:21:44 2021 @author: Paul from PIL import Image import javabridge import bioformats javabridge.start_vm(class_path=bioformats.JARS) import cv2 import matplotlib.pyplot as plt import matplotlib.pyplot as plt import skimage from skimage import data from skimage.filters import threshold otsu import os import numpy as np import skimage.transform as trans import numpy as np import tensorflow from tensorflow.keras.models import * from tensorflow.keras.layers import * from tensorflow.keras.optimizers import * from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler from tensorflow.keras import backend as keras from tensorflow.keras.preprocessing.image import ImageDataGenerator import sklearn.feature extraction from sklearn.datasets import load sample image from sklearn.feature extraction import image from datetime import datetime from keras.utils import to categorical from tensorflow.python.keras.optimizers import * import random import matplotlib.pyplot as plt from mpl toolkits.mplot3d.art3d import Poly3DCollection import numpy as np from skimage import exposure, io, util #%% Specify directory containing training data Training data directory = 'C:/Users/John/20211118 Python biostudies/Python/3D UNET PYTHON/3D UNET PYTHON/1 ImageData/TRAIN/DATA/' # Specify channel number in training data that contains reflectance information RL training directory = 3; # Specify directory containing training labels Training_labels_path = 'C:/Users/John/20211118 Python biostudies/Python/3D UNET PYTHON/3D UNET PYTHON/1 ImageData/TRAIN/LABELS/' ;

```
# COMMIT TRAINING DATA TO DIRECTORY rescaled [0 1] IN MAT FORMAT
MYDIR
='C:/Users/John/20211118 Python biostudies/Python/3D UNET PYTHON/3D UNET PYTHON/
1 ImageData/TRAIN/LABELS/mat training data/'
CHECK FOLDER = os.path.isdir(MYDIR)
# If folder doesn't exist, then create it.
if not CHECK FOLDER:
    os.makedirs(MYDIR)
Training labels holder =
skimage.io.imread('C:/Users/John/20211118 Python biostudies/Python/3D UNET PYTHO
N/3D UNET PYTHON/1 ImageData/TRAIN/LABELS/LABELS 001.tif') ;
Training labels = np.where(Training labels holder == 3, 0,
Training labels holder)
#%%
for i in range(0,2):
    loop = i+1
    counter = '00'+str(loop)
    # find metadata describing file
    TOTAL path = Training data directory+'TRAIN '+counter+'.tif'
    reader = bioformats.ImageReader(TOTAL path)
    NUMBERIMAGES = reader.rdr.getSeriesCount(TOTAL path)
    number of channels = reader.rdr.getSizeC(TOTAL path)
    Xlength = reader.rdr.getSizeX(TOTAL path)
    Ylength = reader.rdr.getSizeY(TOTAL path)
    stackSizeZ = reader.rdr.getSizeZ(TOTAL path)# number of Z slices
    channel zimage = np.zeros((Xlength,Ylength,stackSizeZ))
    # Load reflectance information
    for zplane in range(1,stackSizeZ+1):
        channel zimage[:,:,(zplane-1)] = bioformats.load image(TOTAL path,z =
(zplane-1), c= (RL training directory-1))
    IM DATA = channel zimage
    IM DATA = np.double(IM DATA);
    DIM = IM DATA.shape
    IM DATA norm holder = np.zeros((DIM[0],DIM[1],DIM[2]),np.double)
    IM DATA norm = cv2.normalize(IM DATA, IM DATA norm holder, 1.0, 0.0,
cv2.NORM MINMAX)
    pw = os.getcwd()
    FILENAME = pw+'\\npy training data\\'+'TRAIN DATA '+counter
    FOLDER CHECK = pw+'\\npy training data\\'
    CHECK FOLDER = os.path.isdir(FOLDER CHECK)
    if not CHECK FOLDER:
        os.makedirs(FOLDER CHECK)
    np.save(FILENAME,IM DATA norm)
```

```
pw = os.qetcwd()
#%% COMMIT TRAINING LABELS TO DIRECTORY IN MAT FORMAT
FOLDER NAME = pw+'\\npy training labels\\'
CHECK FOLDER2 = os.path.isdir(FOLDER NAME)
if not CHECK FOLDER2:
    os.makedirs(FOLDER NAME)
for i in range(0,2):
    loop = i + 1
    counter = '00'+str(loop)
    label zimage = [];
    label zimage = np.zeros((Xlength,Ylength,stackSizeZ))
    # load label information
    for zplane in range(1,stackSizeZ+1):
        TOTAL path = Training labels path+'LABELS '+counter+'.tif'
        iplane = bioformats.load image(TOTAL path,z=(zplane-1), c=0,rescale =
False )
        label zimage[:,:,(zplane-1)] = iplane
    IM LABELS = np.double(label zimage);
    DIM = IM LABELS.shape
    IM LABELS norm holder = np.zeros((DIM[0],DIM[1],DIM[2]),np.double)
    IM LABELS norm holder[:,:,:] = label zimage[:,:,:]
    pw = os.getcwd()
    FILENAME = pw+'\\npy_training_labels\\'+'TRAIN_LABELS_'+counter
    FOLDER CHECK = pw+'\\npy training labels\\'
    CHECK FOLDER = os.path.isdir(FOLDER CHECK)
    if not CHECK FOLDER:
        os.makedirs(FOLDER_CHECK)
   np.save(FILENAME, IM_LABELS_norm_holder)
#%%
NUMBER batch = 1
del
     channel zimage, IM LABELS, IM DATA, label zimage
# Prepare to extract patches
patchPerImage = 375;
xrand = random.sample(range(0,959), patchPerImage)
yrand = random.sample(range(0,959), patchPerImage)
zrand = np.random.choice(78,patchPerImage, replace=True)
# Define random patch extraction and augmentarion of patches
def get_train_patched(xrand, yrand, zrand, BATCH_SIZE, batch_number =
NUMBER batch, trainXm=IM DATA norm, trainYm=IM LABELS norm holder):
    patched data image = np.zeros((BATCH SIZE,64,64,32))
    patched data labels = np.zeros((BATCH SIZE,64,64,32))
    for i in range(0,BATCH SIZE):
        val = (i + batch number) - 1;
        patched_data_image[i,:,:,:] =
trainXm[(xrand[val]):(xrand[val]+64),(yrand[val]):(yrand[val]+64),(zrand[val]):(
zrand[val]+32)];
```

```
patched_data_labels[i,:,:,:] =
trainYm[(xrand[val]):(xrand[val]+64),(yrand[val]):(yrand[val]+64),(zrand[val]):(
zrand[val]+32)];
   return patched data image, patched data labels
def get train augmented(trainXm,trainYm ,BATCH SIZE,batch number =
NUMBER batch):
   for i in range(0,BATCH SIZE):
       rand number = random.sample(range(BATCH SIZE), 1)
       my arr = [np.rot90, np.flipud, np.fliplr]
       if int(rand number[0])<3:
          trainXm[i,:,:,:] = my arr[int(rand number[0])](trainXm[i,:,:,:])
          trainYm[i,:,:,:] = my_arr[int(rand_number[0])](trainYm[i,:,:,:])
       elif int(rand_number[0]) ==3:
          trainXm[i,:,:,:] = np.fliplr(trainXm[i,:,:,:])
          trainYm[i,:,:,:] = np.fliplr(trainYm[i,:,:,:])
          trainXm[i,:,:,:] = np.rot90(trainXm[i,:,:,:])
          trainYm[i,:,:,:] = np.rot90(trainYm[i,:,:,:])
       else:
          trainXm[i,:,:,:] = trainXm[i,:,:,:]
          trainYm[i,:,:,:] = trainYm[i,:,:,:]
       return trainXm, trainYm
#88
# CREATE THE 3D UNET
# input layer 64x64x32x1
# encoder depth 4
# 32 filters at the level of the first encoder
def unet3(pretrained weights = None, input size= (64,64,32,1), n class=4):
   inputs = tensorflow.keras.Input(shape=input size)
   conv1 = Conv3D(32, kernel size=(3, 3, 3), dilation rate=2, padding = 'same',
kernel initializer = 'he normal')(inputs)
   conv1 = BatchNormalization()(conv1)
   conv1 = Activation('relu')(conv1)
   conv1 = Conv3D(64, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(conv1)
   conv1 = BatchNormalization()(conv1)
   up1 = Activation('relu')(conv1)
   pool1 = MaxPooling3D(pool size=(2,2,2), strides = (2,2,2))(up1)
   conv2 = Conv3D(64, kernel size=(3, 3, 3), dilation rate=2,padding = 'same',
kernel initializer = 'he normal')(pool1)
   conv2 = BatchNormalization()(conv2)
   conv2 = Activation('relu')(conv2)
   conv2 = Conv3D(128, kernel size=(3, 3, 3), dilation rate=2,padding = 'same',
kernel_initializer = 'he_normal')(up1)
   conv2 = BatchNormalization()(conv2)
   up2 = Activation('relu')(conv2)
   pool2 = MaxPooling3D(pool size=(2,2,2),strides = (2,2,2))(up2)
```

```
conv3 = Conv3D(128, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(pool2)
    conv3 = BatchNormalization()(conv3)
    conv3 = Activation('relu')(conv3)
    conv3 = Conv3D(256, kernel size=(3, 3, 3), dilation rate=2, padding = 'same',
kernel initializer = 'he normal')(conv3)
   conv3 = BatchNormalization()(conv3)
    up3 = Activation('relu')(conv3)
    pool3 = MaxPooling3D(pool_size=(2,2,2),strides = (2,2,2))(up3)
    conv4 = Conv3D(256, kernel size=(3, 3, 3), dilation rate=2,padding =
'same', kernel initializer = 'he normal')(pool3)
   conv4 = BatchNormalization()(conv4)
    conv4 = Activation('relu')(conv4)
    conv4 = Conv3D(512, kernel_size=(3, 3, 3), dilation_rate=2,padding =
'same', kernel initializer = 'he normal')(conv4)
    conv4 = BatchNormalization()(conv4)
    up4 = Activation('relu')(conv4)
    pool4 = MaxPooling3D(pool size=(2,2,2),strides = (2,2,2))(up4)
    conv5 = Conv3D(512, kernel_size=(3, 3, 3), dilation_rate=2,padding =
'same', kernel initializer = 'he normal')(pool4)
   conv5 = BatchNormalization()(conv5)
    conv5 = Activation('relu')(conv5)
    conv5 = Conv3D(1024, kernel_size=(3, 3, 3), dilation_rate=2,padding =
'same', kernel initializer = 'he normal')(conv5)
    conv5 = BatchNormalization()(conv5)
    conv5 = Activation('relu')(conv5)
    drop5 = Conv3DTranspose(1024, kernel_size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation rate=2, padding = 'same', kernel initializer =
'he normal')(conv5)
   merge6 = concatenate([drop5,up4], axis = 4)
    conv6 = Conv3D(512, kernel size=(3, 3, 3), dilation rate=2, padding =
'same', kernel initializer = 'he normal')(merge6)
    conv6 = BatchNormalization()(conv6)
    conv6 = Activation('relu')(conv6)
    conv6 = Conv3D(512, kernel size=(3, 3, 3), dilation rate=2,padding =
'same', kernel initializer = 'he normal')(conv6)
   conv6 = BatchNormalization()(conv6)
    conv6 = Activation('relu')(conv6)
    drop6 = Conv3DTranspose(512, kernel_size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation rate=2, padding = 'same', kernel initializer =
'he normal')(conv6)
   merge7 = concatenate([drop6,up3], axis = 4)
    conv7 = Conv3D(256, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(merge7)
    conv7 = BatchNormalization()(conv7)
    conv7 = Activation('relu')(conv7)
    conv7 = Conv3D(256, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(conv7)
   conv7 = BatchNormalization()(conv7)
    conv7 = Activation('relu')(conv7)
    drop7 = Conv3DTranspose(256, kernel size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation rate=2, padding = 'same', kernel initializer =
'he_normal')(conv7)
   merge8 = concatenate([drop7,up2], axis = 4)
    conv8 = Conv3D(128, kernel size=(3, 3, 3), dilation rate=2, padding =
'same', kernel initializer = 'he normal')(merge8)
```

```
conv8 = BatchNormalization()(conv8)
   conv8 = Activation('relu')(conv8)
    conv8 = Conv3D(128, kernel_size=(3, 3, 3), dilation rate=2,padding =
'same', kernel initializer = 'he normal')(conv8)
   conv8 = BatchNormalization()(conv8)
    conv8 = Activation('relu')(conv8)
    drop8 = Conv3DTranspose(128, kernel size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation_rate=2,padding = 'same', kernel initializer =
'he_normal')(conv8)
    conv8 = BatchNormalization()(drop8)
    conv8 = Activation('relu')(conv8)
    drop8 = Conv3DTranspose(128, kernel_size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation rate=2, padding = 'same', kernel initializer =
'he_normal')(conv7)
   merge9 = concatenate([drop8,up1], axis = 4)
    conv9 = Conv3D(64, kernel size=(3, 3, 3), dilation rate=2,padding = 'same',
kernel_initializer = 'he_normal')(merge9)
   conv9 = BatchNormalization()(conv9)
    conv9 = Activation('relu')(conv9)
   conv9 = Conv3D(64, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(conv9)
   conv9 = BatchNormalization()(conv9)
   conv9 = Activation('relu')(conv9)
   conv9 = Conv3D(n class, kernel size=(1, 1, 1), activation = 'softmax')(conv9)
   model = tensorflow.keras.Model(inputs = inputs, outputs = conv9)
   model.compile(optimizer = Adam(lr = 0.0005),loss =
'sparse_categorical_crossentropy', metrics = ['sparse_categorical_accuracy'])
    if(pretrained weights):
     model=keras.models.load model(pretrained weights)
    return model
#%% Specify location to save the network
checkpoint filepath =
'C:/Users/John/20211118 Python biostudies/Python/3D UNET PYTHON/3D UNET PYTHON/3
Saved models/'
now = datetime.now()
dt string = now.strftime("%d/%m/%Y%H:%M:%S")
NAME =dt string
Name_file = 'PeyersPatchBiostudies '
checkpoint_filepath_name = checkpoint_filepath + Name_file+dt_string
checkpoint dir = os.path.dirname(checkpoint filepath name)
model checkpoint callback = tensorflow.keras.callbacks.ModelCheckpoint(
    filepath=checkpoint dir,
    save_weights_only=True,
   monitor='val accuracy',
   mode='max',
    save best only=True)
#Specify training options
```

```
BATCH SIZE = 8
NUM EPOCHS = 50
patchPerImage = 375;
steps = np.floor(patchPerImage/8);
NUMBER batch = 1;
start = 0;
finish = 8;
storeX = np.zeros((375, 64, 64, 32, 1))
storeY = np.zeros((375, 64, 64, 32, 1))
#Extract patches
for i in range(1, int(steps)):
    [patched data image, patched data labels]=get train patched(xrand, yrand,
zrand, BATCH SIZE, batch number = NUMBER batch, trainXm=IM DATA norm,
trainYm=IM LABELS norm holder)
    [trainXmgg, trainYmgg] =
get train augmented(BATCH SIZE=8, trainXm=patched data image,
trainYm=patched data labels, batch number = NUMBER batch)
    trainYmgg = trainYmgg.reshape((8,64,64,32,1))
    trainXmgg = trainXmgg.reshape((8,64,64,32,1))
    trainXmgg = trainXmgg.astype('float32')
    trainYmgg = trainYmgg.astype('float32')
    storeX[start:finish,:,:,:,:] = trainXmgg
    storeY[start:finish,:,:,:,:] = trainYmgg
    start = finish
    finish = start+8
    NUMBER batch = NUMBER batch+1
[patched data image, patched data labels]=get train patched(xrand, yrand, zrand,
BATCH SIZE=(patchPerImage-finish), batch number = NUMBER batch,
trainXm=IM DATA norm, trainYm=IM LABELS norm holder)
[trainXmgg, trainYmgg] = get train augmented(BATCH SIZE=(patchPerImage-
finish),trainXm=patched data image, trainYm=patched data labels, batch number =
NUMBER batch)
trainYmgg = trainYmgg.reshape((patchPerImage-finish,64,64,32,1))
trainXmgg = trainXmgg.reshape((patchPerImage-finish,64,64,32,1))
trainXmgg = trainXmgg.astype('float32')
trainYmgg = trainYmgg.astype('float32')
storeX[finish:patchPerImage,:,:,:,:] = trainXmgg
storeY[finish:patchPerImage,:,:,:,:] = trainYmgg
model = unet3(pretrained weights = None, input size =
(64,64,32,1),n class=4)#change to 3
```

```
#%% Train the model
history = model.fit(
    storeX,
    storeY,
    batch_size = BATCH_SIZE,
    epochs = NUM_EPOCHS,callbacks=[model_checkpoint_callback])
plt.plot(history.history['sparse_categorical_accuracy'])
plt.title('Model accuracy')
plt.ylabel('accuracy')
plt.ylabel('accuracy')
plt.legend(['train', 'test'], loc='upper left')
plt.savefig('Training_plot.png')
plt.show()
model_cave("C:(Users/John/20211119_Puthen_biestudies/Puthen/3D)
```

```
model.save("C:/Users/John/20211118_Python_biostudies/Python/3D_UNET_PYTHON/3D_UN
ET PYTHON/3 Saved models/my h53D model.h5")
```

% PYTHON SCRIPT: TEST 3D UNET % All code, image-data and screen-cast tutorial videos available for download at % the Biostudies database under accession number S-BSST742 # -*- coding: utf-8 -*-Created on Mon Sep 20 16:37:00 2021 @author: Paul from PIL import Image import javabridge import bioformats javabridge.start vm(class path=bioformats.JARS) import cv2 import matplotlib.pyplot as plt import matplotlib.pyplot as plt import skimage from skimage import data from skimage.filters import threshold_otsu import os import numpy as np import skimage.transform as trans import numpy as np import tensorflow from tensorflow.keras.models import * from tensorflow.keras.layers import * from tensorflow.keras.optimizers import * from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler from tensorflow.keras import backend as keras from tensorflow.keras.preprocessing.image import ImageDataGenerator import sklearn.feature extraction from sklearn.datasets import load sample image from sklearn.feature extraction import image from datetime import datetime from keras.utils import to categorical from tensorflow.python.keras.optimizers import * import random import matplotlib.pyplot as plt from mpl toolkits.mplot3d.art3d import Poly3DCollection import numpy as np from skimage import exposure, io, util from skimage import data from tensorflow.keras.preprocessing.image import ImageDataGenerator import sklearn.feature extraction from sklearn.datasets import load sample image from sklearn.feature extraction import image import matplotlib.cm as cm import matplotlib.cbook as cbook from matplotlib.path import Path from matplotlib.patches import PathPatch import math import json import numpy as np import PIL import imageio

```
#%% Specify directory containing test data
Test data directory =
'C:/Users/John/20211118 Python biostudies/Python/3D UNET PYTHON/3D UNET PYTHON/1
ImageData/TEST/'
# Specify channel number in test data that contains reflectance information
RL channel number = 2;
#COMMIT REFLECTANCE TEST DATA TO DIRECTORY RESCALED [0 1] IN MAT FORMAT
MYDIR
='C:/Users/John/20211118_Python_biostudies/Python/3D UNET PYTHON/3D UNET PYTHON/
4 TEST Unet/npy test data/'
CHECK FOLDER = os.path.isdir(MYDIR)
# If folder doesn't exist, then create it.
if not CHECK FOLDER:
    os.makedirs(MYDIR)
for i in range(0,1):
    loop = i+1
   counter = '00'+str(loop)
   TOTAL path = Test data directory+'TEST '+counter+'.tif'
   # find metadata describing file
   reader = bioformats.ImageReader(TOTAL path)
   NUMBERIMAGES = reader.rdr.getSeriesCount(TOTAL path)
   number of channels = reader.rdr.getSizeC(TOTAL path)
   Xlength = reader.rdr.getSizeX(TOTAL path)
    Ylength = reader.rdr.getSizeY(TOTAL path)
    stackSizeZ = reader.rdr.getSizeZ(TOTAL path)
   channel zimage = np.zeros((Xlength,Ylength,stackSizeZ))
    # Load reflectance information
    for zplane in range(1,stackSizeZ+1):
        channel zimage[:,:,(zplane-1)] = bioformats.load image(TOTAL path,z =
(zplane-1), c= (RL channel number-1))
    IM_DATA = channel_zimage
    IM DATA = np.double(IM DATA);
    DIM = IM DATA.shape
    IM DATA norm holder = np.zeros((DIM[0],DIM[1],DIM[2]),np.double)
    IM DATA norm = cv2.normalize(IM DATA, IM DATA norm holder, 1.0, 0.0,
cv2.NORM MINMAX)
    pw = os.getcwd()
    FILENAME = pw+'\\npy_test_data\\'+'TEST_DATA '+counter
    FOLDER_CHECK = pw+'\\npy_test_data\\'
    CHECK FOLDER = os.path.isdir(FOLDER CHECK)
    if not CHECK FOLDER:
        os.makedirs(FOLDER CHECK)
   np.save(FILENAME,IM_DATA_norm)
```

```
#88
#Redefine network
def unet3(pretrained weights = None, input size= (64,64,32,1), n class=4):
    inputs = tensorflow.keras.Input(shape=input size)
    conv1 = Conv3D(32, kernel_size=(3, 3, 3), dilation_rate=2, padding = 'same',
kernel initializer = 'he normal')(inputs)
   conv1 = BatchNormalization()(conv1)
    conv1 = Activation('relu')(conv1)
    conv1 = Conv3D(64, kernel size=(3, 3, 3), dilation rate=2,padding = 'same',
kernel initializer = 'he normal')(conv1)
   conv1 = BatchNormalization()(conv1)
    up1 = Activation('relu')(conv1)
    pool1 = MaxPooling3D(pool_size=(2,2,2),strides = (2,2,2))(up1)
    conv2 = Conv3D(64, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(pool1)
   conv2 = BatchNormalization()(conv2)
   conv2 = Activation('relu')(conv2)
    conv2 = Conv3D(128, kernel_size=(3, 3, 3), dilation rate=2,padding = 'same',
kernel initializer = 'he normal')(up1)
   conv2 = BatchNormalization()(conv2)
    up2 = Activation('relu')(conv2)
   pool2 = MaxPooling3D(pool size=(2,2,2),strides = (2,2,2))(up2)
    conv3 = Conv3D(128, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel_initializer = 'he_normal')(pool2)
    conv3 = BatchNormalization()(conv3)
    conv3 = Activation('relu')(conv3)
    conv3 = Conv3D(256, kernel size=(3, 3, 3), dilation rate=2,padding = 'same',
kernel initializer = 'he normal')(conv3)
   conv3 = BatchNormalization()(conv3)
    up3 = Activation('relu')(conv3)
   pool3 = MaxPooling3D(pool size=(2,2,2), strides = (2,2,2))(up3)
   conv4 = Conv3D(256, kernel_size=(3, 3, 3), dilation_rate=2,padding =
'same', kernel initializer = 'he normal')(pool3)
    conv4 = BatchNormalization()(conv4)
    conv4 = Activation('relu')(conv4)
    conv4 = Conv3D(512, kernel size=(3, 3, 3), dilation rate=2,padding =
'same', kernel initializer = 'he normal')(conv4)
   conv4 = BatchNormalization()(conv4)
    up4 = Activation('relu')(conv4)
   pool4 = MaxPooling3D(pool_size=(2,2,2),strides = (2,2,2))(up4)
   conv5 = Conv3D(512, kernel_size=(3, 3, 3), dilation_rate=2,padding =
'same', kernel initializer = 'he_normal')(pool4)
   conv5 = BatchNormalization()(conv5)
   conv5 = Activation('relu')(conv5)
    conv5 = Conv3D(1024, kernel_size=(3, 3, 3), dilation_rate=2,padding =
'same', kernel initializer = 'he normal')(conv5)
    conv5 = BatchNormalization()(conv5)
    conv5 = Activation('relu')(conv5)
    drop5 = Conv3DTranspose(1024, kernel_size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation_rate=2,padding = 'same', kernel_initializer =
'he normal')(conv5)
   merge6 = concatenate([drop5,up4], axis = 4)
    conv6 = Conv3D(512, kernel size=(3, 3, 3), dilation rate=2,padding =
'same', kernel initializer = 'he normal')(merge6)
   conv6 = BatchNormalization()(conv6)
    conv6 = Activation('relu')(conv6)
    conv6 = Conv3D(512, kernel size=(3, 3, 3), dilation rate=2,padding =
'same', kernel initializer = 'he normal')(conv6)
```

```
conv6 = BatchNormalization()(conv6)
   conv6 = Activation('relu')(conv6)
   drop6 = Conv3DTranspose(512, kernel_size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation_rate=2, padding = 'same', kernel_initializer =
'he normal')(conv6)
   merge7 = concatenate([drop6,up3], axis = 4)
   conv7 = Conv3D(256, kernel size=(3, 3, 3), dilation rate=2, padding = 'same',
kernel_initializer = 'he_normal')(merge7)
   conv7 = BatchNormalization()(conv7)
   conv7 = Activation('relu')(conv7)
   conv7 = Conv3D(256, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(conv7)
   conv7 = BatchNormalization()(conv7)
   conv7 = Activation('relu')(conv7)
   drop7 = Conv3DTranspose(256, kernel size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation rate=2, padding = 'same', kernel initializer =
'he normal')(conv7)
   merge8 = concatenate([drop7,up2], axis = 4)
   conv8 = Conv3D(128, kernel_size=(3, 3, 3), dilation_rate=2,padding =
'same', kernel initializer = 'he normal')(merge8)
   conv8 = BatchNormalization()(conv8)
   conv8 = Activation('relu')(conv8)
   conv8 = Conv3D(128, kernel size=(3, 3, 3), dilation rate=2,padding =
'same', kernel initializer = 'he_normal')(conv8)
   conv8 = BatchNormalization()(conv8)
   conv8 = Activation('relu')(conv8)
   drop8 = Conv3DTranspose(128, kernel size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation rate=2, padding = 'same', kernel initializer =
'he normal')(conv8)
   conv8 = BatchNormalization()(drop8)
   conv8 = Activation('relu')(conv8)
    drop8 = Conv3DTranspose(128, kernel_size=(2, 2, 2), strides=(2, 2, 2),
activation = 'relu', dilation rate=2, padding = 'same', kernel initializer =
'he normal')(conv7)
   merge9 = concatenate([drop8,up1], axis = 4)
   conv9 = Conv3D(64, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(merge9)
   conv9 = BatchNormalization()(conv9)
   conv9 = Activation('relu')(conv9)
   conv9 = Conv3D(64, kernel_size=(3, 3, 3), dilation_rate=2,padding = 'same',
kernel initializer = 'he normal')(conv9)
   conv9 = BatchNormalization()(conv9)
   conv9 = Activation('relu')(conv9)
   conv9 = Conv3D(n class, kernel_size=(1, 1, 1), activation = 'softmax')(conv9)
```

```
model = tensorflow.keras.Model(inputs = inputs, outputs = conv9)
model.compile(optimizer = Adam(lr = 0.0005),loss =
'sparse categorical crossentropy', metrics = ['sparse categorical accuracy'])
```

```
if(pretrained_weights):
     model=keras.models.load model(pretrained weights)
   return model
#%% load the trained 3D Unet network
#model.load weights("C:/Users/John/20211118 Python biostudies/Python/3D UNET PYT
HON/3D_UNET_PYTHON/3_Saved_models/my_h53D_model.h5")
pretrained model = unet3()
pretrained model.load weights("C:/Users/John/20211118 Python biostudies/Python/3
D UNET PYTHON/3D UNET PYTHON/3 Saved models/my h53D model updated.h5")
#%% Patch the reflectance data through the network
# Set the patch sizes to be passed to the net
patchSize = [64, 64, 32]
# Segment blockwise then reassemble in full
# Define image dimensions
Shape overall = np.shape(IM DATA norm)
height = Shape overall[0]
width = Shape overall[1]
depth = Shape overall[2]
if len(Shape overall) == 3:
   nChannel = 1
else:
   nChannel = Shape overall[3]
patch = np.zeros([patchSize[0],patchSize[1],patchSize[2],
nChannel],dtype=IM DATA norm.dtype);
number_of_height_patches = math.ceil(height/patchSize[0]);
number of width patches = math.ceil(width/patchSize[1]);
number_of_depth_patches = math.ceil(depth/patchSize[2]);
#Pad image to have dimensions as multiples of patchSize
height_pad = number_of_height_patches*patchSize[0];
width pad = number of width patches*patchSize[1];
depth pad = number of depth patches*patchSize[2];
# Amount to pad different dimensions of image
padSize = np.empty([1, 3])
# Pad the image by correct amounts
padSize[0,0] = height pad;
padSize[0,1] = width_pad;
padSize[0,2] = depth_pad;
Starting_shape = np.shape(IM_DATA_norm)
a1 = padSize[0,0]
a2 = padSize[0,1]
a3 = padSize[0,2]
im_pad = np.zeros([int(a1),int(a2),int(a3)])
im_pad[:IM_DATA_norm.shape[0],:IM_DATA_norm.shape[1],:IM_DATA_norm.shape[2]] =
IM DATA norm[:,:,:]
```

```
im_pad = np.asarray(im_pad).reshape((im_pad.shape[0],
im pad.shape[1],im pad.shape[2],1))
[height pad, width pad, depth pad, nChannel pad] = np.shape(im pad);
#Preallocate some matrices to catch the probability maps
out_Uncertainty_Scores = np.zeros([height_pad, width_pad, depth_pad],'double');
out scores from network all classes = np.zeros([height pad, width pad,
depth_pad, 4], 'double');
#Loop through blocks of 'patchSize'
for loop height in range(1,number of height patches+1):
    for loop width in range(1, number of width patches+1):
        for loop depth in range(1, number of depth patches+1):
            PP3 = np.empty([1, 64, 64, 32, 1])
            start height position=(loop height-1)*patchSize[0];
            end height position=loop height*patchSize[0];
            start width position=(loop width-1)*patchSize[1];
            end_width_position=loop_width*patchSize[1];
            start depth position=(loop depth-1)*patchSize[2];
            end depth position=loop depth*patchSize[2];
patch to deploy=im pad[start height position:end height position,start width pos
ition:end width position,start depth position:end depth position,:];
            PP3[0,:,:,:]=patch to deploy
            # deploy net
            predictions = pretrained model.predict(PP3)
            scores = np.max(predictions,axis=4)
            predictions.argmax(axis=3)
```

out_Uncertainty_Scores[start_height_position:end_height_position,start_width_pos ition:end_width_position,start_depth_position:end_depth_position]=scores

out_scores_from_network_all_classes[start_height_position:end_height_position,st
art_width_position:end_width_position,start_depth_position:end_depth_position,0]
=predictions[0,:,:,:,0];

out_scores_from_network_all_classes[start_height_position:end_height_position,st
art_width_position:end_width_position,start_depth_position:end_depth_position,1]
=predictions[0,:,:,:,1];

```
out_scores_from_network_all_classes[start_height_position:end_height_position,st
art_width_position:end_width_position,start_depth_position:end_depth_position,2]
=predictions[0,:,:,:,2];
```

```
out_scores_from_network_all_classes[start_height_position:end_height_position,st
art_width_position:end_width_position,start_depth_position:end_depth_position,3]
=predictions[0,:,:,:,3];
```

```
#Script is general for n classes to this point
out Pmap cat1 = out scores from network all classes[:,:,:,0];
out Pmap cat2 = out scores from network all classes[:,:,1];
out Pmap cat3 = out scores from network all classes[:,:,2];
out Uncertainty Scores = out Uncertainty Scores[0:height, 0:width, 0:depth];
#Remove padding from probability maps
out Pmap cat1 = out Pmap cat1[0:height, 0:width, 0:depth];
out Pmap cat2 = out Pmap cat2[0:height, 0:width, 0:depth];
out_Pmap_cat3 = out_Pmap_cat3[0:height, 0:width, 0:depth];
#%%visualise network outputs
fig, (ax1, ax2, ax3, ax4) = plt.subplots(1, 4, sharey=True)
fig.set size inches(14, 12)
im = ax1.imshow(out Pmap cat1[:,:,59])
ax1.title.set text('Background/Other')
im = ax2.imshow(out_Pmap_cat2[:,:,59])
ax2.title.set_text('LF-nuclei')
im = ax3.imshow(out Pmap cat3[:,:,59])
ax3.title.set text('LF-actin')
im = ax4.imshow(out Uncertainty Scores[:,:,59])
ax4.title.set_text('Uncertainty')
plt.savefig('Predictions.png')
#%%Map to 16-bit and save for loading into CellProfiler
data_norm_holder = np.zeros((1024, 1024, 107), np.uint16)
uil6_PMAP_cat1_xy = cv2.normalize(out_Pmap_cat1, data_norm_holder , 65535.0,
0.0, cv2.NORM MINMAX)
data_norm_holder = np.zeros((1024, 1024, 107), np.uint16)
uil6 PMAP cat2 xy = cv2.normalize(out Pmap cat2, data norm holder , 65535.0,
0.0, cv2.NORM MINMAX)
data_norm_holder = np.zeros((1024, 1024, 107), np.uint16)
uil6 PMAP cat3 xy = cv2.normalize(out Pmap cat3, data norm holder , 65535.0,
0.0, cv2.NORM MINMAX)
data to save = np.zeros((1024,1024,107,3))
data to save[:,:,:,0] = ui16 PMAP cat1 xy
data to save[:,:,:,1] = ui16 PMAP cat2 xy
data_to_save[:,:,:,2] = ui16_PMAP_cat3_xy
#%% write files out as multipage tiff
for channel in range(0,3):
    image to save = (data to save[:,:,:,channel]).astype(dtype=np.uint16)
    rearranged image to save = np.transpose(image to save, axes=[2,0,1])
imageio.mimwrite('Ch '+str(channel+3)+' lm 001'+'.tiff',rearranged image to sav
e)
```

Extracting single-cell features using label-free cell segmentation and CellProfiler 4

- Download and install CellProfiler4 from the CellProfiler website. https://cellprofiler.org



1. 2-D CellProfiler Pipeline

-A screencast video is included with the BioStudies archive.

- Folder 5 "5_CellProfiler_cell_measurements" in the 2-D BioStudies project archive contains example data, a CellProfiler pipeline and the subsequent CellProfiler single-cell outputs:





- The image-data (inside 1_Structured_test_data) contains four immunofluorescence channels followed by the probability maps obtained for the label-free nuclei, actin and background/other classes from either the Python or MATLAB deep learning scripts (channels 5, 6 and 7, respectively). - Open CellProfiler 4 and load the image analysis pipeline from the "2_CellProfiler_pipeline" folder.

- Drag-and-drop the "1_structured_test_data" folder into the CellProfiler image-window to load the 2-D image-data:

Image: Signal Control State View	ments		- 0	× ⊕ ⊕ ⊕	er 4.1.5: CM15_PeyensPatch lestData lest Windows Help \$ Images \$ Metadata \$ NamesAndTypes \$ Groups \$ Threshold \$ Threshold \$ Threshold \$ MaskImage	Channels - page to contract much and any encloses minuted out over the contract of the contrac				
Image: Second secon	Move Copy to* Copy Organise NatLAB_2D_UNET > 5_CellProfiler_co	New item * Easy access * Properties Call ew Open ell_measurements v C	n • ESelect all Select none ory Invert selection Select Select	0 G 0 G 0 G 0 G 0 G 0 G 0 G 0 G 0 G 0 G	5 Maskimage 5 RescaleIntensity 6 RescaleIntensity 6 GrayToColor 5 GrayToColor 5 SaveImages 5 SaveImages	Dron files and folders	hore			
This PC 30 Objects Desktop Bocuments Documents	Name 1,Structured,test_data 2,CellProfiler_pipeline 3,Outputs	Date modified 30/11/2021 14:08 30/11/2021 14:08 30/11/2021 14:09	Type Sid File folder File folder File folder	(ze) Φ G G G Φ G G G Φ G G G Φ G G G Φ G G G Φ G G G Φ G G G Φ G G G G Φ G G G G G Φ G	5 Savemages 5 Threshold 5 Maskimage 6 ImageMath 5 GaussianFilter 5 Threshold 5 ImageMath 6 Savemathelat	Drop mes and loiders	liere			
Music Videos Videos Videos Videos Videos Videoss Video	• <			0000 0000 0000 0000 0000 0000 0000 0000 0000	Maskimage (Maskimage) (Maskimage) Sovermages Sovermages OverlayOutlines Savelmages (MeasureObjectSizeShape MeasureObjectIntensity	Filter imaget [®] [No filtering	?			

- Choose where to save the outputted cell features by clicking on the "Output Settings" tab at the bottom-left of the CellProfiler dialogue.


- To run the CellProfiler pipeline and save the cell features and other pipeline outputs at the specified location, click "Analyse Images" at the bottom-left of the CellProfiler screen.

CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellC	Dutlines.cpproj (D:\John\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell —	×
CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellC File Edit Test Windows Help C Images C Metadata C NamesAndTypes C Groups C Threshold C Threshold C MaskImage C RescaleIntensity C RescaleIntensity C GrayToColor C Gray	Dutlines.cpproj (D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_ - Channel_1 = CD11c Channel_3 = DF Channel_4 = lymphoid tissue mask Channel_5 = label-free actin probability map from Unet Channel_6 = label-free actin probability map from Unet Channel_7 = OD1:d Channel_6 = label-free actin probability map from Unet Channel_7 = OD1:d Channel_7 = Divlohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_1_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_3_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_3_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_5_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_6_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_6_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_6_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1_Structured_tet Ch_6_m_001:df D:Vohn\2021\MATLAB_biostudies\MATLAB_2D_UNET\5_CellProfiler_cell_measurements\1	× it_data\/ it_data\/ it_data\/ it_data\/ it_data\/ ? ?
Output Settings View Workspace		
? Adjust modules: + - ^ V I Start Test Mode Analyze Images	Found 7 rows	

- Previously-saved CellProfiler outputs are also available inside the "3_Outputs" folder:

MATLAB_biostudie	s > MATLAB_2D_UNET > 5_CellProfiler_ce	ell_measurements v	🖸 🔎 Se				
^	Name	Date modified Typ					
	1_Structured_test_data	30/11/2021 14:08	File folder				
	2_CellProfiler_pipeline	30/11/2021 14:08	File folder				
	3_Outputs	30/11/2021 14:09	File folder				

2. 3-D CellProfiler Pipeline

-A screencast video is included with the BioStudies archive.

- Folder 5 "5_CellProfiler_cell_measurements" in the 3-D BioStudies project archive contains example data, a 3-D CellProfiler pipeline and the subsequent CellProfiler single-cell outputs:

□ < DATA (D:) > John > 2021	> MATLAB_biostudies > MATLAB_3D_UNET	5_CellProfiler_cell_measu	rements ~	ē
^	Name	Date modified	Туре	Size
ts		30/11/2021 14:10	File folder	
	2_CellProfiler_pipeline	30/11/2021 14:10	File folder	
nts	3_Outputs	30/11/2021 14:10	File folder	



- The image-data (inside 1_Structured_test_data) contains two immunofluorescence channels followed by the 3-D probability maps obtained for the label-free nuclei, actin and background/other classes from either the Python or MATLAB deep learning scripts (channels 3, 4 and 5, respectively).

- Open CellProfiler 4 and load the image analysis pipeline from the "2_CellProfiler_pipeline" folder.

- Drag-and-drop the "1_structured_test_data" folder into the CellProfiler image-window to load the 2-D image-data:



- Choose where to save the outputted cell features by clicking on the "Output Settings" tab at the bottom-left of the CellProfiler dialogue.

Ce	ellProfile	r 4.1.3: CP413	3_3D_MLN_pipeline.cpproj (E	:\John\2021\MATLAB_biostudies\MATLAB_3D_UNET\5_CellProfiler_cell_measurements\2_CellPro	_		×
File	Edit Te	st Window	vs Help				
	C	Images		2 Default land Falder cillilluscretillelaire			
	S	Metadata					
	c	NamesAnd	Types	? Default Output Folder: D:\\CELLPROFILER_OUT		\sim	
	S	Groups					
	<⊅ ₪	Threshold					
	<⊅ ₪	Maskimage	2				
	<⊅ ₪	Savelmage	s				
	<⊅ ₪	Savelmage	s				
	<₽ €	Watershed					
	<₽ €	ConvertOb	jectsTolmage				
	<₽ €	GaussianFil	ter				
	(1)	Threshold					
	Image: Second	ImageMath	1				
	Image: Second	RemoveHo	les				
	(1)	Watershed					
		ConvertOb	Jects lolmage				
		Savelmage	S				
		MassureOb	ojectintensity				
		EvportToSp	ojectoizeonape				
	\$P 🕒	Exportiosp	readsheet				
	Outp	ut Settings	View Workspace				
	outh	- seconda	Ten ronspore				
?	Adj	ust modules	+ - ^ v				
Ð	Start Te	st Mode	Analyze Images				

- To run the CellProfiler pipeline and save the cell features and other pipeline outputs at the specified location, click "Analyse Images" at the bottom-left of the CellProfiler screen.

CellProfiler 4.1.3: CP413_	3D_MLN_pipeline.cpproj ([D:\John\2021\MATLAB_biostudies\MATLAB_3D_UNET\5_CellProfiler_cell_measurements\2_CellPro — 🛛 💫	<
File Edit Test Windows	Help		
Images File Edit Test Windows Images Images Images Metadata Images Images Images Images <tdimages< td=""> Images</tdimages<>	3D_MLN_pipeline.cpproj (I Help /pes ctsToImage sr ctsToImage ectIntensity ectSizeShape addsheet	D:\John\2021\MATLAB_biostudies\MATLAB_3D_UNET\5_CellProfiler_cell_measurements\2_CellPro — — — — — — — — — — — — — — — — _ > >	
🗇 🗹 ExportToSpre	eadsheet	 ✓ Show files excluded by filters 	>?
		Filter images? Images only ? Apply filters to the file list ?	
Output Settings	View Workspace		
2 Adiational di	nan nonspace		
Adjust modules: Start Tart Made	+ · · · V]	
Start Test Mode	Analyze Images	Found 5 rows	

- Previously-saved CellProfiler outputs are also available inside the "3_Outputs" folder:

A w DATA (D:) > Joł	hn → 2021 :	> MATLAB_biostudies > MATLAB_3D_UNET >	5_CellProfiler_cell_measure	ments v	5
	^	Name	Date modified	Туре	Size
ts			30/11/2021 14:10	File folder	
		2_CellProfiler_pipeline	30/11/2021 14:10	File folder	
nts		3_Outputs	30/11/2021 14:10	File folder	8

2-D Cell Profiler image analysis pipelines. This section presents screenshots of a CellProfiler image analysis pipeline used to achieve label-free cell segmentation in 2-D from the Unet network outputs, and to measure the intensity and size/shape features of identified cell-objects. To use the image analysis pipeline with new image data, the 'IdentifyPrimaryObjects' module simply needs adjusting so that the 'typical diameter of objects' size-range matches the pixel scaling of the new images. For newcomers to CellProfiler, we recommend downloading the image-data and pipeline from BioStudies database https://www.ebi.ac.uk/biostudies/ under accession number S-BSST742. This enables the pipeline to be run with the data described in the manuscript and allows the user to see how each module works.

Image: Bit The Workers Hege Image: Bit The Workers Hege <th>CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell</th> <th>Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FRE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch</th> <th>. 🗆 X</th>	CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell	Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FRE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch	. 🗆 X
Channel - CD12 Chane	File Edit Test Windows Help		
Marinda Montal (1 - CD1) Convert 3: 9 Wind Markel Convert	⊠ Images		
Vancakaflyse George Googe Goo	Metadata	Channel_1 = CD11c	^
Concept Section C	NamesAndTypes	Channel_2 = CD3	
	Groups	Channel 3 = UP	
	() C Threshold	Channel 5 = label-free nuclei probability man from Unet	
Constraining	() C Threshold	Channel 6 = label-free actin probability map from Unet	
Sustainage Sustaina	() C Maskimage		
 C. L. I., MO.1.8 C. Saccialization M. C. L. I., MO.1.8 C. L. I., MO.1.8 C. L. I., MO.1.8 C. L. I., MO.1.8 C. Saccialization M. C. L. I., MO.1.8 C. L. I., MO.1.8 C. Saccialization M. C. M. MO.1.20, LABL, FREL SCIANTIATION, PROJECTS 2021032, 2D, Sactude A.J., Fail J. Depar. Patch assign MATLAB.2D, PyrenPatch S, Cell Profile cell measurements 1, S. C. M. MO.1.8 C. Saccialization M. M. M. MO.1.20, LABL, FREL SCIANTIATION, PROJECTS 2021032, 2D, Sactude A.J., Fail J. Depar. Patch assign MATLAB.2D, PyrenPatch S, Cell Profile cell measurements 1, S. C. M. MO.1.80, D. PyrenPatch S, Cell Profile cell measurements 1, S. C. M. MO.1.80, D. PyrenPatch S, Cell Profile cell measurements 1, S. C. M. MO.1.80, D. PyrenPatch S, Cell Profile cell measurements 1, S. C. Saccialization M. M. MO.2000, MO.1.20, LABL, FREL SCIANTIATION, PROJECTS 2021032, 2D, Sactude A.J., Fail J. Depar. Patch assign MATLAB.2D, PyrenPatch S, Cell Profile cell measurements 1, S. C. Saccialization M. M. MO.2000, MO.1.20, LABL, FREL SCIANTIATION, PROJECTS 2021032, 2D, Sactude A.J., Fail J. Depar. Patch assign MATLAB.2D, PyrenPatch S, Cell Profile cell measurements 1, S. C. Saccialization M. C. Saccialization M. M. MO.2000, MO.1.20, LABL, FREL SCIANTIATION, PROJECTS 2021032, 2D, Sactude A.J., Fail J. Depar. Patch assign MATLAB.2D, PyrenPatch S, Cell Profile cell measurements 1, S. C. Saccialization M. C. M. MO. MO. MO. MO.2000, MO.1.20, LABL, FREL SCIANTIATION, PROJECTS 2021032, 2D, Sactude M. T. M. J. M. MO. MO. MO.2000, MO.1.2000, MO.1.2	Maskimage	D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch\5_CellProfiler_cell	_measurements\1_St
		Ch_1_lm_001.tif	
 Constructions Construc		D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch\5_CellProfiler_cell	_measurements\1_St
 Operation of the comparison of the	() C GravInColor	Ch_2_Im_U01.tdt D_01.bbs/2020/2020/417_2D_1_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_MATLAD\2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APEL_EPEE_SEGMENTATION_DROLECTS\20210522_2D_biostudies\2_Eixel_2D_Dware_Databased_APE	massuraments\1 St
 S. Starmage S. Starm	() C GravToColor	Ch 3 im 00.titi	_measurements(1_5
 Church Cliff Churc	() Savelmager	D:/John/2020/20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS/20210522_2D_biostudies/2_Final 2D Peyers Patch example/MATLAB/2D_PeyersPatch/5_CellProfiler_cell	_measurements\1_St
 C Tueshal C Tueshal C Mukanoge Mukanoge <li< td=""><td>Savelmages</td><td>Ch_4_Im_001.tif</td><td></td></li<>	Savelmages	Ch_4_Im_001.tif	
 Control Mathemate Control Mathemate<	C Devenhoges C Threshold	D:\John\2020/20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch\5_CellProfiler_cell	_measurements\1_St
 C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. S. Jim, 2014/fi C. Jim, 2014/fi S. SaveInage: SaveInage: SaveInage: SaveInage: SaveInage:	🔿 🗹 Maskimage	Cn_2_IIII_WILLIIII EI IN/JANN/2020/20200417 2D LABEL FREE SEGMENTATION DROJECTS/20210522 2D biostudies/2 Final 2D Devers Datch example/MATLAR/2D DeversDatch/S CellDrofiler cell	measurements\1 St
		Ch 6 Im 001.iff	_measurements/1_3
 Churchold Chur		D: John/2020/2020/417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch\5_CellProfiler_cell	_measurements\1_St
© E inspektati, © MasuroDigit: © Adjust module: • • • • • • • • • • • • • • • • • • •	♂ S Threshold	Ch_7_Im_001.tiff	
Oudput Setting: Vew Workspace 2 Adjust module: Post Trat Mode Found 1 image sets	♂ S ImageMath		,
Image: Stating: Sta		•	
IdentifyFirmsyObjects IdentifyFirmsyObjec	Maskimage		3
Interinger, Konnethig Interinger, Konnethig Sectors and Secto		Eilter imager? No fileging	2
© E Savetmags © Sovetmags © Sovetmags © MesureObjetciteShape © EbportToSpredsheet Output Settings Yee Workspace 2 Adjuar modules: + + Savet Test Mode Found 1 image sets		niteringes: No filtering V	f
© E Overlag Outlines © E MessureObjectIntensity © E ExportToSpreadsheet			
 ♥ G Savelinger ♥ G MesureObjectSizeShape ♥ G EponToSpreadsheet 			
 [®] G MasureObjectSizeShape [®] G ExportToSpreadtheet [®] G ExportToSpreadtheet [®] G ExportToSpreadtheet [®] G Adjust modules [®] G G G G G G G G G G G G G G G G G G G			
G KessureObjectintensity G ExportToSpreadsheet Uutput Settings View Workspace Adjust modules: + - ^ v Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules • Start Test Mode • Analyze Images			
Output Settings View Workspace 2 Adjust modules + - + - + Start Test Mode + Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - * Start Test Mode * Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - +> Start Test Mode -> +> Start Test Mode >> +> Start Test Mode >>			
Output Settings View Workspace ? Adjust modules: + - > Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - I b Start Test Mode I mage sets			
Output Settings View Workspace 2 Adjust modules: + - I > Start Test Mode > Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - > Start Test Mode Found 1 image sets	1		
Output Settings View Workspace ? Adjust modules: + - > Start Test Mode Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ v > Start Test Mode > Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ * * * * * * Start Test Mode * Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - > Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ v > Start Test Mode > Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ v Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ > Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - > Start Test Mode > Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ > Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ v I Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ V Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - > Start Test Mode Analyze Images Found 1 image sets			
Output Settings View Workspace ? Adjust modules: + - ^ v Start Test Mode Analyze Images Found 1 image sets			
? Adjust modules: + - ^ V > Start Test Mode > Analyze Images Found 1 image sets	Output Sattings View Washerson		
Start Test Mode Analyze Images Found 1 image sets	? Adjust modules: + - ^ v		
Found 1 image sets	Ib Stat Task Made		
	Analyze Images	Found 1 image sets	



CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellC	Outlines.cpproj* (D:	\John\2020\202	00417_2D_LABEL	FREE_SEGMENTA	TION_PROJECTS	\20210522_2D_bio	studies\2_Final 2D	Peyers Patch exar	mple\MATLAB\2D_PeyersPatch.	. –		×
File Edit Test Windows Help												
🕑 Images												_
🖾 Metadata	Loads images a	and associates t	hem together acc	cording the specif	ied rule criteria							\sim
☑ NamesAndTypes												
🖸 Groups												
International 🗇 🗹 Threshold												
< 🗹 Threshold												\sim
< 🗹 Maskimage												
🗇 ビ Maskimage	Assic	n a name to In	nages matching r	ules 🗸							?	^
🗇 🗹 RescaleIntensity	-											
I RescaleIntensity	Pr	ocess as 3D?									?	
I GrayToColor												
I GrayToColor		Ma	atch All 🗸 of	the following rule	5						?	
I Savelmages	Select the	e rule criteria —	Disastana		. A . L .	Char	nel 1				i	
I Savelmages			Directory ~		ntain	~ Chan	nei_i			- +		
🗇 🗹 Threshold	Name to assign t	hese images c	1								?	
🔿 🗹 Maskimage												
🔿 🗹 ImageMath	Select the	e image type Gr	rayscale image	\sim							?	
🔿 🖸 GaussianFilter	Set intensity	range from Im	nage bit-depth	~							?	
G Inreshold												
C Imagewath		L	Juplicate this ima	ige								
Kentovenores												
IdentifyPrimaryObjects		Ma	atch All 🗸 of	the following rule	s						?	
ConvertObjectsTolmage	Select the	e rule criteria —	D : 1			C					i	
			Directory ~	Does V Coi	ntain	~ Chan	nei_2			- +		
I CverlayOutlines	Name to assign t	hese images 🥻	2								?	
🗇 🗹 Savelmages												
I MeasureObjectSizeShape	Select the	e image type Gi	rayscale image	~							3	
I MeasureObjectIntensity	Set intensity	range from Im	nage bit-depth 🕚	~							?	
I ExportToSpreadsheet			Qualicata this ima								2	
			Supreate this inte	ige							· ·	
		F	Remove this imag	je							?	
		Ma	atch All 🗸 of	the following rule	<.						2	
	Select the	e rule criteria				Char					i	
		_	Directory ~	Does V	ntain	~ Chan	nei_5			- +		
	Name to assign t	hese images c	3								?	
								-			-	
	Update	cl	C2	C3	c4	co	CD	c/				
	1 (Ch_1_lm_001.tif	Ch_2_Im_001.tif	Ch_3_Im_001.tif	Ch_4_Im_001.tif	Ch_5_Im_001.tiff	Ch_6_Im_001.tiff	Ch_7_Im_001.tiff				
Output Settings View Workspace												
? Adjust modules: + - ^ v												
Start Test Mode Analyze Images	Found 7 rows											
	.,											_



CellProfile	er 4.1.3: CP413	PeyersPatchTestData_CellC	Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch	×
File Edit T	est Window	rs Help		
C	Images			
C	Metadata		I his module thresholds the ci i mages (CU i i c channe) The output is a binary image for civels above the threshold	
C	NamesAnd	Types		
C	Groups			
۵ 🖾	Threshold			
Image: Contract of the second seco	Threshold			× .
	Maskimage	1		
	Recoleinte	e situ	Select the input image c1 v (from NamesAndTypes)	?
0 0 0	RescaleInte	nsity	Name the systematic typesh CD11c binary	2
() C	GravToCol	or		•
ا ا	GrayToCol	or	Threshold strategy Global 🗸	?
ي 🖒	Savelmage	5	Thresholding method Manual	2
ا ال	Savelmage	5		•
ا 🖾	Threshold		Manual threshold 0.01	?
<	Maskimage	:	Threshold smoothing scale 0	?
S	ImageMath	1		•
<!<!<!<!<!<!<!<!<!<!<!<!<!<!<!<</td <td>GaussianFil</td> <td>ter</td> <td></td> <td></td>	GaussianFil	ter		
	Ihreshold			
	Pamageiviatr	l los		
	Mackimage	ies .		
	ldentifvPrir	narvObiects		
() C	ConvertOb	iectsTolmage		
() C	Savelmage	5		
ا 🖉	0verlayOut	lines		
ا 🖉	Savelmage	5		
ى 🖒	MeasureOb	jectSizeShape		
ا 🖉	MeasureOb	jectIntensity		
< ♥ ☑	ExportToSp	readsheet		
0	aut Cattings	View Workspace		
2	liust modules	view workspace		
r Ad	just modules	+ - ~ V		
Start T	est Mode	Analyze Images	Found 7 rows	

Comparison The module thresholds the clamage (CD3 channe) Comparison The output is a binary image for pilots above the threshold Comparison Statustication Statustication Statustic	it Test Windows Help	
Output Interaction	C Images	This module thresholds the c2imaoes (CD3 channel)
Company product The endpoint is a binary image for picks above the threshold Company product The endpoint is a binary image for picks above the threshold Company product Security Security Security	Nietadata	
Comparison Second Second Second <	Groups	The output is a binary image for pixels above the threshold
Computational Status Image Status Computational Status Status Status		
Comput Stating View Workspace Appler Stating View Workspace Appler Stating View Workspace	C Threshold	
Composition Select the input image Image: Imag	⊕ 🖸 Maskimage	
Control Status Contr	🗇 🖸 Maskimage	Select the input impage 2 (from Namer And Turger)
Corpor Setting: Yeer Workspace Apport Setting: Yeer Workspace	🗇 🗹 RescaleIntensity	
Corput Stating: Yee Workspace: Apjust mediate: Image:	🗇 ビ RescaleIntensity	Name the output image thresh_CD3_binary
Corput Setting: Verw Workspace Agust module: • • • • • • • • • • • • • • • • • • •	🗇 🗹 GrayToColor	
Copper Setting: Verw Workspret Apjurt module: •	I GrayToColor 🖸 🗇	Inreshold strategy Global V
Corpert Setting: VerwWorkspace: Adjust models: • • • • • • • • • • • • • • • • • • •	I Savelmages	Thresholding method Manual 🗸
Output Stating: VerwWorkspace: Adjust mednes: • • • • • • • • • • • • • • • • • • •	🗇 🗹 Savelmages	
Output Stating: VerwWorkspace Agust models: • • • • • • • • • • • • • • • • • • •	🗇 🗹 Threshold	Manual threshold 0012
Output Stitling: Verw Vorkpace Adjut modules: <!--</td--><td>🗇 🗹 Maskimage</td><td>Threshold smoothing scale</td>	🗇 🗹 Maskimage	Threshold smoothing scale
Output Stetling: View Workspace: Adjut module: Image Nath	🗇 🗹 ImageMath	
Output Setting: View Workspace Adjust module: Notes	🗇 🕒 GaussianFilter	
Output Settings View Workspace Adjut modules • • • • • • • • • • • • • • • • • • •	Inresnoid	
Output Setting: View Workspace Adjust modules: • Adjust modules: • Adjust modules: • Versupport • Versupport • Versupport • Versupport • •<	🗇 🖸 Imageiviath	
Output Settings View Workspace Adjust modules: • • • • • • • • • • • • • • • • • • •	🖤 🖸 Kemoverioles	
Output Setting: View Workspace Adjut module: • Adjut module: • Very Workspace • • •	🖤 🖸 Maskimage	
Output Settings View Workspace Adjust modules:	GenuryPrimaryObjects ConvertObjectsTolmage	
Output Setting: View Workspace Adjust module: •	Convertobjects ronnage Savelmager	
Output Settings View Workspace Adjust modules: • • •	OverlayOutlines	
 Vew Workspace Adjust modules A v 	C Savelmages	
© E kportToSpreadsheet Output Settings View Workspace Adjust modules + + ∧ ∨	MeasureObjectSizeShape	
© EpontToSpreadsheet		
Output Settings View Workspace Adjust modules: + A V	🗇 ビ ExportToSpreadsheet	
Output Settings View Workspace Adjust modules: + - ^ V		
Output Settings View Workspace Adjust modules: + + . N V		
Output Settings View Workspace Adjust modules:		
Output Settings View Workspace Adjust modules: + • v		
Output Settings View Workspace Adjust modules: <lu> </lu> <lu> </lu> 		
Output Settings View Workspace Adjust modules: •		
Output Settings View Workspace Adjust modules: +		
Output Settings View Workspace Adjust modules: +		
Output Settings View Workspace Adjust modules: • • • • •		
Output Settings View Workspace Adjust modules: + - ^ v		
Output Settings View Workspace Adjust modules: + - ^ ^ v		
Output Settings View Workspace Adjust modules: + - ^ ^ v		
Output Settings View Workspace Adjust modules: + - ^		
Output Settings View Workspace Adjust modules: + - ^ V		
Output Settings View Workspace Adjust modules: + - ^		
Output Settings View Workspace Adjust modules: + - ^ / v		
Output Settings View Workspace Adjust modules: + - ^ ^ v		
Output Settings View Workspace Adjust modules: + - ^		
Output Settings View Workspace Adjust modules: + - ^ v		
Adjust modules: + - ^ v	Output Settings View Works	ace
Adjust modules: + - ^ v		
	Adjust modules: + -	

CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellC	Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FRE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch – 📃	×
File Edit Test Windows Help		
Mages		
Metadata	This module creates a greyscale version of the c1 (CD11c) data - where the threshold calculated above is applied.	^
NamesAndTypes		
Groups	Ine outputted image from this module can be measured in each cell-object to calculate per-cell intensity values AF LEX the background has been thresholded out. This is particulary imported here as the importance of this impacts in call object and layer for background corrections according to the intensity of a second avoid to the adjust of the second avoid to the	
	important nele, as integration of this image in centrologicus anows for background conection according to the dissue-matched secondary-only and sotype controls.	
C Threshold		~
↓ S Maskimage		2
	Select the input image c1 (from NamesAndTypes)	- ?
	Next the set of these h CD11c groups all	2
CravEcolor		
() C GrayToColor	Use objects or an image as a mask? Image V	?
() C Glayrocolor		
() C Savelmages	Select image for mask thresh_CD11c_binary v (from Threshold #05)	?
G Saveinages		2
A C Maskimage	Invert the mask? O Yes No	- f
G ImageMath		
() C GauccianFilter		
() C ImageMath		
C RemoveHoles		
Kentovenoies		
IdentifyPriman(Objects		
ConvertObjects		
Savelmager		
() Saveninges		
Swelmager		
MeasureObjectSizeShape		
MeasureObjectIntensity		
W C Export rospreadsneet		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode Analyze Images	Found 7 rows	

🔽 CellProfi	iler 4.1.3: CP413_PeyersPatchTestData_Cell	Outlines.cpproj* (D:\John\2020\202004	17_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch –	×						
File Edit	Test Windows Help									
C	🕉 Images	This module creates a groupsale u	arrian of the x2 (CD2) data , where the threshold calculated above is applied							
0	Metadata This induce cleates a greyscale version of the C2 (CD3) data - where the threshold calculated above is applied.									
	NamesAndTypes	The outputted image from this m	tputted image from this module can be measured in each cell-object to calculate per-cell intensity values AFTER the background has been thresholded out. This is particulary ant here, as integration of this image in cell objects allows for background correction according to the tissue-matched secondary-only and isotype controls.							
	S Groups	important here, as integration of t	here, as integration of this image in cell objects allows for background correction according to the tissue-matched secondary-only and isotype controls.							
90	Threshold									
() ()	Maskimage									
0	3 Maskimage			2						
٠	S RescaleIntensity	Select the input image	c2 (from NamesAnd Types)	-						
ا ال	S RescaleIntensity	Name the output image	thresh_CD3_greyscale	?						
ه⊳ ۵	GrayToColor									
ا 🕫	GrayToColor	Use objects or an image as a mask?	Image V	?						
⊲> 0	Savelmages	Select image for mask	thresh CD3 binary v (from Threshold #06)	?						
(1)	Savelmages	-								
(2)	S Inreshold	Invert the mask?		?						
() () () ()	S ImageMath									
d (GaussianFilter									
۵ C	3 Threshold									
ا (ا	🖸 ImageMath									
ا لا	RemoveHoles									
۞ (\$	🛛 Maskimage									
ۍ ک	IdentifyPrimaryObjects									
(1)	ConvertObjectsTolmage									
(2)	Savelmages									
() ()	Savelmager									
¢۲ د ه د	MeasureObjectSizeShape									
٢.	MeasureObjectIntensity									
<	S ExportToSpreadsheet									
0	tout Cattings View Workspace									
U	input settings view workspace									
? A	Adjust modules: + - ^ v									
Start 🕨	Test Mode Analyze Images	Found 7 rows								

CellProfil	er 4.1.3: CP413_PeyersPatchTestData_Cell	Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —	
File Edit T	Test Windows Help		
6	images Metadata	This module rescales the 12-bit thresholded CD11c data for subsequent visulisation and saving.	^
C	7 NamesAndTypes 7 Groups	Saturated pixels (i.e., 4095) in the source 12-bit image are rescaled to the maximum	
() ()	Threshold		U.
\$ C	Maskimage		
() C	Maskimage	Select the input impact that CD11 - moved and (from Mackimans #07)	2
() C	RescaleIntensity	Select the input image these Contegreyscale (find washinge +07)	
Image: A marked block in the second secon	RescaleIntensity	Name the output image thresh_CD11c_greyscale_Rescaled	?
ا 🗇	GrayToColor	Pression mathematical field and the second sec	
() C	GrayToColor	Rescaining method Choose specific values to be reset to a custom range	<u> </u>
() C	Savelmages	Method to calculate the minimum intensity Custom V	?
0 C	5 Saveimages 7 Threshold	Method to calculate the maximum intensity Custom	?
0	Maskimage		
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	/mageMath	Intensity range for the input image 0.0 0.0025	ſ
() C	GaussianFilter	Intensity range for the output image 0.0 1.0	?
ا 🖉	7 Threshold		
Image: A marked block in the second secon	/ ImageMath		
<⇒ €	RemoveHoles		
	Maskimage		
	ConvertObjects		
0 C	Savelmages		
1 C	0verlavOutlines		
ا 🔁	Savelmages		
Image: A marked black in the second secon	MeasureObjectSizeShape		
() C	MeasureObjectIntensity		
() C	ExportToSpreadsheet		
Out	put Settings View Workspace		
? Ac	djust modules: + - ^ v		
Note: T	est Mode 🛛 🕨 Analyze Images	Found 7 rows	

CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell	Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FRE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —	
File Edit Test Windows Help		
C Images		
G Metadata	This module rescales the 12-bit thresholded CD3 data for subsequent visulisation and saving.	~
NamesAndTypes	Saturated pixels (i.e. 4005) in the source 12-bit image are servaled to the maximum	
🖾 Groups	Saturated pixels (i.e., 405) in the source iz-bit image are rescaled to the maximum	
🗇 🗹 Threshold		
Interstold		\sim
🗇 🗹 Maskimage		
🗇 🗹 Maskimage	Select the input image thresh_CD3_greyscale <pre>v (from MaskImage #08)</pre>	?
RescaleIntensity		
KescaleIntensity	Name the output image Thresh_CU3_greyscale_Kescaled	1
	Rescaling method Choose specific values to be reset to a custom range 🗸 🗸	?
	Mathead to as leader the minimum intensity for the second s	2
		£
🗇 🕑 Threshold	Method to calculate the maximum intensity Custom V	?
🗇 🗹 Maskimage	Intensity range for the input image 0.0 0.0625	?
ImageMath		
🔿 🗹 GaussianFilter	Intensity range for the output image 0.0 1.0	3
🗇 🗹 Threshold		
🔿 🕑 ImageMath		
Kemovenoies		
IdentifyPrimaryObjects		
I Savelmages		
🗇 🗹 OverlayOutlines		
I Savelmages		
I MeasureObjectSizeShape		
I MeasureObjectIntensity		
ExportToSpreadsheet		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode Analyze Images	Found 7 rours	
	Irouna / rows	

File Edit Tes	st Windows Help		
। । । । । ।	Images Metadata NamesAndTypes Groups	This module makes a colour image of the thresholded CD11c image data	<
@ U	I hreshold		
	Threshold		~
\$ 5	Maskimage		_
	Maskimage	Select a color scheme Composite \checkmark ?	
0 C	RescaleIntensity	Name the output image View thresh CD11c rescaled ?	
	GravTeColor		- 1
	GravToColor	Rescale intensity Over One	
0	Savelmages	lected intensity O yes @No	
\$ C	Savelmages	made name threeh CD11s exercise Recorded or (from PerceleIntentity #00)	
1	Threshold	image name tutest_CDTTC_greystate_rescaled v (non-rescalentensisy +09)	- 1
S	Maskimage	Color	
Image: A marked black of the second secon	ImageMath	w : u 10	- I
🗇 🕑	GaussianFilter	weight 10	
I	Threshold	Add another channel Add another channel ?	
₫ 🗇	ImageMath		-
S	RemoveHoles		
< ♥ ♥	Maskimage		
Image: Second	IdentifyPrimaryObjects		
	ConvertObjects I olmage		
0 C	Saveimages		
	Savelmages		
0	MeasureObjectSizeShane		
00	MeasureObjectIntensity		
0 5	ExportToSpreadsheet		
Outpu	ut Settings View Workspace		
? Adju	ust modules: + - ^ v		
Start Tes	st Mode 🕨 Analyze Images	Found 7 rows	

CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellOutlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_final 2D Peyers Patch example\MATLAB\2D_PeyersPatch... -

CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell	IIOutlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch	– 🗆 ×
File Edit Test Windows Help		
び Images び Metadata び NamesAndTypes	This module makes a colour image of the thresholded CD3 image data	^
Groups		<u>_</u>
 	Select a color scheme Composite V	?
 	Name the output image View_thresh_CD3_rescaled Rescale intensity O Yes No	?
 	Image name thresh_CD3_greyscale_Rescaled v (from RescaleIntensity #10)	?
🗇 🗹 Maskimage	Color	?
	Weight 1.0	?
 	Add another channel Add another channel	?
 ⑦ E RemoveHoles ⑦ E Maskimage ⑦ E IdentifyPrimaryObjects ⑦ E ConvertObjectsTolmage ⑦ E SaveImages 		
Export I obpreadsheet Export I o		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode Analyze Images	Found 7 rows	



CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellOutlines.cpproj* (D:\John\2020/20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch... -



🔽 CellProfiler 4.1.3: CP413_PeyersPa	tchTestData_CellOutline	es.cpproj* (D:\John\2020\20200417_	2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —		×
File Edit Test Windows Help					
☑ Images ☑ Metadata ☑ NamesAndTypes ☑ Groups ☑ Threshold ④ ☑ ☑ Threshold	Cr	reates a binary mask of the image-r	egion containing lymphoid tissue		<
🚸 🗹 Maskimage					
🗇 🗹 Maskimage		Select the input image	c4 v (from NamesAndTypes)	1	?
🗇 🗹 RescaleIntensity					
KescaleIntensity KescaleIntensity		Name the output image	Iymphoid_tissue_KOI_mask		<i>(</i>
GrayToColor		Threshold strategy	Global 🗸	1	?
		Thresholding method	Oteu		2
I Savelmages		in caloung neurod			•
🗇 🗹 Threshold	Two	o-class or three-class thresholding?	Two classes 🗸	1	?
I Maskimage		Threshold smoothing scale	0	1	?
🗇 🕑 ImageMath		Theorem and a second second second	10		2
		Infeshold correction factor			:
	Low	ver and upper bounds on threshold	0 1.0	1	?
🗇 🕑 RemoveHoles					?
I Maskimage	L	.og transform before thresholding?	○ Yes ◉ No		
🚸 🗹 IdentifyPrimaryObject	ts				
ConvertObjects Folma	age				
OverlayOutlines					
🚸 🕑 MeasureObjectSizeSh	ape				
I MeasureObjectIntens	ity				
I ExportToSpreadsheet					
Output Settings View Wo	rkspace				
? Adjust modules: +	- ^ v				
Start Test Mode	nalyze Images Four	ind 7 rows			

File Edit Test Windows Help	
Images Applies the lymphoid ROI mask to the LF-actin probabil NamesAndTypes Groups Threshold Fraction probability	ility map used to enable cell-segmentation
🗇 🕑 Threshold	
🗇 🗹 Maskimage	
	(from NamesAndTypes) ?
	?
🗇 🗹 GrayToColor	
	<i>₹</i>
	Il mask v (from Threshold #15) ?
	?
Maskimage	
() Gaussianhilter	
() C Inteshold	
Maskimane	
IdentifyPrimaryObjects	
ConvertObjectsTolmage	
🛷 🗹 MeasureObjectIntensity	
🗇 🕑 ExportToSpreadsheet	
Output Settings View Workspace	
? Adjust modules: + - ^ v	
Start Test Mode Analyze Images Found 7 rows	

© Threshold © Threshold © Makalange © Makalange © Stackalitationally © Stackalitationaly © Stackalitationa	び Metadata び NamesAndTypes び Groups	Inverts the LF-actin probability ma	o such that the 'cytoplasm' of each cell appears as segmentable foreground
Current Nationage Compositional State	Croups Croups Threshold	-	
Output Stating Operation wrett	⊕ G Threshold		
Operation Interdet	🗇 🖸 Maskimage		
Current Stream The state stream Current Stream Name the output single inverted_marked_d Current Stream Stream Current Stream Stream Current Stream Making the first image bill Current Stream Stream	🗇 🖸 Maskimage	Operation	Invert V
Control Section	🗇 🗹 RescaleIntensity	operation	
Output Stating: Verw Workspace Output Stating: Verw Workspace	🗇 🗹 RescaleIntensity	Name the output image	inverted_masked_c6
Option Setting: Yee Workspeet Option Setting: Yee Workspeet	🗇 🗹 GrayToColor		
© Sevenages Setext the first image Makked_6 (from Maskimge #16) © Sevenages Multiply the first image Makked_6 (from Maskimge #16) © Maskimage Multiply the first image by 10 © Gustamilier Multiply the first image by 10 © Gustamilier Multiply the result by 10 © Gustamilier Multiply the result by 10 © Gustamilier Multiply the result by 10 © Sevenages Set values less than 0 equal to 07 © Sevenages Set values less than 0 equal to 17 © MessureObjectificEnhage Set values less than 0 equal to 17 © MessureObjectificEnhage No @ MessureObjectificEnhage No @ MessureObjectificEnhage No @ MessureObjectificEnhage No <tr< td=""><td>🗇 🗹 GrayToColor</td><td></td><td></td></tr<>	🗇 🗹 GrayToColor		
© Sevenages Multiply the first image by © Convertigies Multiply the result by © Convertigies Set values less than 0 equal to 0? © Soverlages Set values greater than 1 equal to 1? © MeasureDigetSintennagy Set values greater than 1 equal to 1? © Soverlages Set values greater than 1 equal to 1? © Soverlages Set values greater than 1 equal to 1? © Soverlages Set values greater than 1 equal to 1? © Soverlages Set values greater than 1 equal to 1? © Soverlages Set values greater than 1 equal to 1? © Soverlages Set values greater than 1 equal to 1? © Soverlages Set values greater than 1 equal to 1? © Soverlages No Set subjectimenagy Set values with 0? © Soverlages No Set subjectimenagy Set © No Ignore the image masks? Yes @ No	🗇 🗹 Savelmages	Select the first image	Masked_c6 v (from MaskImage #16)
Output Setting: View Workspace: Output Setting: View Workspace:	I Savelmages	Multiplushed Statistics and built	10
© MaskImage © Constrainting © Constrainting <td>Ihreshold</td> <td>Multiply the first image by</td> <td>10</td>	Ihreshold	Multiply the first image by	10
Cotput Setting Yeew Worksace Adjust modules +	🗇 🗹 Maskimage		
Output Setting: View Workspace: Adjust module: +	🗇 🗹 ImageMath	Deire the second discount is built	10
Output Settings View Workspace Output Settings View Workspace	🗇 ビ GaussianFilter	Raise the power of the result by	
Output Settings: View Workspace Adjust module: +	🗇 🕑 Threshold	Multiply the result by	1.0
Output Settings: View Workspace Adjust module: I	🗇 🗹 ImageMath		
Output Setting: View Workspace: Adjust module: +	🗇 🗹 RemoveHoles	Add to result	0.0
Output Setting: View Workspace Adjust modules: +	🗇 🗹 Maskimage		
Output Setting: View Workspace Adjust modules: +	GentifyPrimaryObjects	Set values less than 0 equal to 0?	● Yes ○ No
Output Setting: View Workspace Adjust modules: +	ConvertObjects Loimage		
Output Settings View Workspace Adjust modules: +	Saveimages Saveimages	Catural and a sector three 1 and 12	
Output Settings View Workspace Adjust modules: +	C Swelmager	Set values greater than 1 equal to 1:	● Yes ○ No
Output Settings View Workspace Adjust modules: +	Saveimages MeasureObjectSizeShape		
Output Settings View Workspace Adjust modules: +		Replace invalid values with 0?	
Output Settings View Workspace Adjust modules: 	Kessureobjeenmensky ExportToSpreadsheet		
Output Settings View Workspace Adjust modules: +		Ignore the image masks?	
Output Settings View Workspace Adjust modules: + - ^ v		ignore the image masts.	V Yes (I) No
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: + - ^			
Output Settings View Workspace Adjust modules: + - ^ v			
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: + - ^ v			
Output Settings View Workspace Adjust modules: +			
Output Settings View Workspace Adjust modules: + - ^ v			
Output Settings View Workspace Adjust modules: + - ^			
Output Settings View Workspace Adjust modules: + - ^ v			
Output Settings View Workspace Adjust modules: + - ^			
Output Settings View Workspace Adjust modules: + - ^			
Output Settings View Workspace			
Adjust modules: + - ^ V v			
	Output Settings View Workspace	_	



File Edit Test Windows Help			
ⓒ Images ⓒ Metadata ⓒ NamesAndTypes ⓒ Groups	Thresholds the smoothed background probability map to yield a binary image		^
 ♥ E Threshold ♥ E Threshold 			~
 		Γ	
♦ ♥ RescaleIntensity	Select the input image Smoothed_background_pmap v (from	i GaussianFilter #18)	7
 	Name the output image binary_background		?
	Threshold strategy Global \sim		?
	Thresholding method Minimum Cross-Entropy 🗸		?
 ♦ Saveimages ♦ E Threshold 	Threshold smoothing scale 0.0		?
 	Threshold correction factor 1.0		?
♦ C Imagenation ♦ C GaussianFilter	ower and upper bounds on threshold 0.0 1.0		?
			?
	Log transform before thresholding? O Yes No	L	-
IdentifyPrimaryObjects ConvertObjectsTolmage			
🚸 🖸 OverlayOutlines			
I Savelmages			
I MeasureObjectSizeShape			
(1) C Export i ospreadsheet			
Output Settings Way Walsones			
2 Adjust modules:			
ID Shed Test Made			
P Start Test Mode Analyze Images	ound 7 rows		

Ē

C mages			
☑ Metadata ☑ NamesAndTyp ☑ Groups	pes	Switches the foreground and back	ground i
🗇 🗹 Maskimage			
🗇 🖸 Maskimage		Operation	Invert 🗸
C RescaleIntensity RescaleIntensity	ty	Name the output image	inverted_binary_background
	y .	1 3	
I Savelmages		Select the first image	binary_background v (from Threshold #19)
Savelmages Savelmages		Multiply the first image by	1.0
		r	
I GaussianFilter		Raise the power of the result by	1.0
		Multiply the result by	1.0
TimageMath			00
		Add to result	
🚸 🗹 IdentifyPrimary	yObjects	Set values less than 0 equal to 0?	OV-CON-
	sTolmage		I res () No
Savelmages Savelmages		Caturalization that 1 and to 12	
Savelmages	5	Set values greater than 1 equal to 1:	Yes O No
🗄 🗹 MeasureObject	tSizeShape	ſ	
🗇 🗹 MeasureObject	tIntensity	Replace invalid values with 0?	● Yes ○ No
I Export To Spread	dsheet		
		Ignore the image masks?	○ Yes ◉ No
Output Settings Vie	iew Workspace		
Output Settings Vie Adjust modules:	iew Workspace		

CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell	Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —	×
File Edit Test Windows Help		
🖸 Images		
🖸 Metadata	Removes small holes from the background other image	~
NamesAndTypes		
Groups		
🔿 🖸 Threshold		~
🗇 ビ Maskimage		
🗇 🖸 Maskimage	Select the input impage invested binner bedramand and (from impace/14th #20)	2
I RescaleIntensity	Select the input image invertee_binary_background (infinitiagewath #20)	
I RescaleIntensity	Name the output image deholed_inverted_binary_background	?
I GrayToColor		
I GrayToColor	Size of holes to fill 20.0	?
I Savelmages		
I Savelmages		
🗇 🗹 Threshold		
🗇 🗹 Maskimage		
💿 🗹 ImageMath		
🗇 🗹 GaussianFilter		
🔿 🖸 Threshold		
Imageiviatn G		
Kemoveholes		
IdentifyBriman/Objects		
ConvertObjects		
Savelmages		
OverlavOutlines		
Savelmages		
I MeasureObjectIntensity		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode		
Printiges	Found 7 rows	

CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellC	Outlines.cpproj* (D:\John\2020\202004	17_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —	X
File Edit Test Windows Help			
🕑 Images	[
🕑 Metadata	Subtracts the background from th	e LF probability map	^
NamesAndTypes			
🖾 Groups			
Inreshold			
Inreshold			\sim
🗇 🗹 Maskimage	-		
🗇 🗹 Maskimage	Select the input image	inverted masked c6 v (from ImageMath #17)	?
🗇 🗹 RescaleIntensity		interce_mosce_co	
I RescaleIntensity	Name the output image	background_masked_seeds	?
I GrayToColor	Use albiests as an image as a mark?		
	Use objects of an image as a mask:	image ~	f
	Select image for mask	deholed_inverted_binary_background v (from RemoveHoles #21)	?
🔿 🖸 Savelmages			
G Ihreshold	Invert the mask?		?
Maskimage Maskimage		U TES ©INO	
Gaussian Filter			
ImageMath			
C RemoveHoles			
🗇 🗹 Savelmages			
International State (1997) International State (
🗇 🗹 Savelmages			
I MeasureObjectSizeShape			
I MeasureObjectIntensity			
I ExportToSpreadsheet			
Output Settings View Workspace			
? Adjust modules: + - ^ v			
Start Test Mode Analyze Images	Found 7 rows		

CellProfiler 4.1.3: CP413_PeyersPatchTestData_CellC	Dutlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FRE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —	
File Edit Test Windows Help		
🖾 Images	I shall free call instance composition using the outputted probability many from the Unit model	
Metadata	Laber-nee cen instance segmentation using the outputted probability insps from the oner model	
☑ NamesAndTypes		
Groups		
< 🗹 Threshold		
💿 🗹 Threshold		×
🔿 🗹 Maskimage		
🔿 🖸 Maskimage	lise advanced settings?	?
Transity	O'Yes O'No	
The scale intensity		2
T Gray I o Color	Select the input image background_masked_seeds v (from Maskimage #22)	1
Gray I o Color	Name the primary objects to be identified Cells	?
Saveimages		
Saveimages	Typical diameter of objects, in pixel units (Min,Max) 26 59	?
() C Maskingang		2
() C Image Math	Discard objects outside the diameter range? 🔿 Yes 💿 No	•
() Constant		
		?
() C ImageMath	Discard objects touching the border of the image? O Yes No	
Mackimage	I hreshold strategy Global V	?
IdentifyPrimaryObjects	Thresholding method Manual 🗸	?
© ConvertObjectsTolmage		
Savelmages	Manual threshold 0.05	7
	Threshold smoothing scale 22	?
	Method to distinguish clumped objects Intensity 🗸	?
	Method to draw dividing lines between clumped objects listencity	2
	include to and arrangement champed objects internet v	•
	Automatically calculate size of smoothing filter for declumping?	?
	Adomatically calculate size of shootaling method accompany. In vest O No	
		2
	Automatically calculate minimum allowed distance between local maxima? 🔿 Yes 💿 No	•
	Suppress local maxima that are closer than this minimum allowed distance 13	?
	Speed up by using lower-resolution image to find local maxima?	?
		?
	Display accepted local maxima? 🔿 Yes 💿 No	
	Fill holes in identified objects? After declumping only	?
	Handling of objects if excessive number of objects identified Continue	?
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode		
p childres p ranayze images	Found 7 rows	

		~
CellProfiler 4.1.3: CP413_PeyersPatch TestData_Cell	Jutlines.cpproj* (D:\John\2020\2020\41/_2D_LABEL_FKE_SEGMENTATION_PKOJECTS\2021D522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —	×
File Edit Test Windows Help		
🖾 Images	This would as some the identified all a birth into a income (with d 'CallMarket) when a solution in the source and a source and the source at the intervent is the income field.	
🕑 Metadata	This module converts the identified cell-objects into an image (called Celliviasis) where each pixel is humbered according to the cell-object humber it belongs to in that image-field.	\sim
NamesAndTypes	This image is useful for colouring specific cell-objects for downstream visulisations of the data e.g., immunofluorescence / nearest-cell neighbour visulisations etc.	
Groups		
Inreshold		
Inreshold		~
🔿 🗹 Maskimage		
🔿 🗹 Maskimage	Select the input objects Cells v (from IdentifyPrimaryObjects #23)	?
KescaleIntensity		
CrayToColor	Name the output image Celliviasks	1
	Select the color format uint 16 v	?
Savelmages		
🗇 ビ Maskimage		
🗇 🗹 ImageMath		
🗇 🗹 GaussianFilter		
Interstold		
ImageMath 🗹		
I RemoveHoles		
🗇 🗹 Maskimage		
ConvertObjectsTolmage	4	
Saveimages		
G Savennages		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode		
Analyze Images	Found 7 rows	



CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell	Outlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers I	Patch example\MATLAB\2D_PeyersPatch —	
File Edit Test Windows Help			
☑ Images ☑ Metadata	This module overlays the label-free cell segmentation onto a chosen image - here the CD3 immunofluorescence data		^
 NamesAndTypes Groups 	In this way, it allows a visual check of how well the cell segmentation is performing.		
🗇 🗹 Maskimage			
			?
KescaleIntensity KescaleIntensity	Vispidy oddinies on a blank integer Ves Ves		
GravToColor	Select image on which to display outlines View thresh CD3 rescaled v (from GravToColor #12)		?
I GrayToColor			
I Savelmages	Name the output image Segmentation_checker		?
I Savelmages	Outline display mode Color 🗸 🗸		?
Inreshold Mackimage	How to outline longer		2
🗇 🗹 GaussianFilter			
< 🗹 Threshold	Select objects to display Cells v (from IdentifyPrimaryObjects #23)		?
	Select outline color		?
KemoveHoles Maskimage	Add another outline		2
	Add shother oddine		•
I ConvertObjectsTolmage			
I Savelmages			
OverlayOutlines			
G Savennages MeasureObjectSizeShape			
I ExportToSpreadsheet			
Output Settings View Workspace			
? Adjust modules: + - ^ v			
Start Test Mode Analyze Images	Found 7 rows		



CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell	IOutlines.cpproj* (D:\John\2020\20200417_2D_LABEL_FREE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch —	
File Edit Test Windows Help		
C Images C Metadata C NamesAndTypes C Groups Ø C Threshold	This module measures shape and size properties for cell-objects	^
		~
🗇 🗹 Maskimage		
🗇 🗹 Maskimage	Cells (from IdentifyPrimaryObjects #23)	?
I RescaleIntensity		
	Select object sets to measure	
() C Savelmager		
🗇 ビ Maskimage		
🗇 ビ ImageMath	Calculate the Torrike for the real of the control of the	?
< 🗹 GaussianFilter	Calculate the Zerlinke Features: O Yes No	
🔿 🗹 Threshold		2
G Remove Holes	Calculate the advanced features? O Yes No	
Maskimage		
< 🗹 ConvertObjectsTolmage		
I Savelmages		
MeasureObjectSizeShape	•	
ф С офинистр		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode Analyze Images	Found 7 rows	
	,	

CellProfiler 4.1.3: CP413_PeyersPatchTestData_Cell	utlines.cpproj* (D:\John\2020\2020417_2D_LABEL_FRE_SEGMENTATION_PROJECTS\20210522_2D_biostudies\2_Final 2D Peyers Patch example\MATLAB\2D_PeyersPatch		×
File Edit Test Windows Help			
G Metadata	This module measures intensity properties in cell-objects for each of the images specified		^
NamesAndTypes			
Groups	e.g., amount of CDTIc or CD3 expression in each cell object		
			~
🔿 🗹 Maskimage			
🔿 🖸 Maskimage			2
	CellMasks (trom ConvertObjects loimage #24)		:
	Sampatrian chacker (from Owerschuld Hollen #16)		
() C GravToColor	Select images to measure a segmentation_checker (inform OverlayOutlines #20)		
	Select mage to measure		
	View thresh CD3 rescaled (from GravToColor #12)		
	background masked seeds (from MaskImage #22)	~	
🔿 🗹 Maskimage			
	Cells (from IdentifyPrimaryObjects #23)		?
🗇 🕑 GaussianFilter	(initial and initial and initiana and initial and initial and initial and initial and init		
🗇 🗹 Threshold			
🚸 🖸 ImageMath	Select objects to measure		
🗇 ビ RemoveHoles			
🚸 🖸 Maskimage			
IdentifyPrimaryObjects			
I ConvertObjectsTolmage			
I Savelmages			
International States (Second Second S			
I Savelmages			
I MeasureObjectSizeShape			
🗇 🗹 MeasureObjectIntensity			
I ExportToSpreadsheet			
Output Settings View Workspace			
Start Test Mode Analyze Images	Found 7 rows		_



3-D CellProfiler pipeline. This section presents screenshots of a CellProfiler image analysis pipeline used to achieve label-free cell segmentation in 3-D from the Unet network outputs and to measure the intensity and size/shape features of identified cell-objects. To use the image analysis pipeline with new image data, the 'IdentifyPrimaryObjects' module simply needs adjusting so that the 'typical diameter of objects' size-range matches the pixel scaling of the new images. For newcomers to CellProfiler, we recommend downloading the image-data and pipeline from BioStudies database https://www.ebi.ac.uk/biostudies/ under accession number S-BSST742. This enables the pipeline to be run with the data described in the manuscript and allows the user to see how each module works.

CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj (x:John\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\ 🛛 🗖	×
File Edit Test Windows Help		
 CellProfiler 4.1.3: CP413_30_MLN_pipeline.cpproj (0 File Edit Test Windows Help Gimages Metadata NamesAndTypes Groups Chreshold Gaselinages SaveImages GaussianFilter ConvertObjectsToImage GaussianFilter ElmageMath RemoveHoles SaveImages SaveImages E SaveImages E SaveImages SaveImages E ConvertObjectsToImage SaveImages E Massimage E KenoveHoles E KenoveHoles E SaveImages 	Auchin 2020 2020017, 30_LABEL_FREE_SEGMENTATION_PROJECTS/20210525_biostudies/2_Final 3D MLN example/MATLABI3D_MLNS_CellProfiler_cell_measurements/1_Structures (channel 3 = FOXP3 channel 4 = LF-Actin probability map channel 4 = LF-Actin probability map chan	X i_test_dat i_test_dat i_test_dat i_test_dat i_test_dat ? ? ?
Output Settings View Workspace ? Adjust modules: + - ^ v ID Start Test Mode Analyze Images	Found 5 rows	



CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj (D:\John\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\... Х

_
File Edit Test Windows Help	:Vonn/2020/2020041/_3D_LABEL_FREE_SEGMENTATION_PROJECTS/20210525_biostudies/2_Final 3D MLN example/MATLAB/3D_MLN/55_CellProfiler_cell_measurements/ —	
☑ Images ☑ Metadata ☑ NamesAndTypes ☑ Groups ④ ☑ Threshold ④ ☑ MaskImage	The NamesAndTypes module allows you to assign a meaningful name to each image by which other modules will refer to it.	< >
 	Assign a name to Images matching rules ?	^
 	Relative pixel spacing in X 0.207	
 	Relative pixel spacing in Y 0.207 ? Putting rind participation in Z 0.150 2	
 ♥ ☑ ConvertObjects I olmage ♥ ☑ SaveImages ♥ ☑ MeasureObjectIntensity 	Match All v of the following rules ?	
 	Metadata v Does v Have channel matching v 1 - + Name to assign these images c1 ?	
	Select the image type Grayscale image V ? Duplicate this image ?	ľ
	Match All of the following rules ? Metadata V Does V Have channel matching V 2 - +	
	Name to assign these images c2 ? Select the image type Gravscale image	
	Duplicate this image ? Remove this image ?	
	Select the rule criteria Match All v of the following rules ? Metadata v Does v Have channel matching v 3 - +	~
	Update c1 c2 c3 c4 c5 1 Ch_1_m_001.tiff Ch_2_lm_001.tiff Ch_3_lm_001.tiff Ch_4_lm_001.tiff Ch_5_lm_001.tiff	
Output Settings View Workspace		
Start Test Mode Analyze Images	Found 5 rows	

r.



CellProfil	er 4.1.3: CP413	_3D_MLN_pipeline.cpproj (E	3:Vohn/2020/20200417_3D_LABEL_FRE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\ — 🛛	×
File Edit T	Fest Window	s Help		
	Images		This module creates a binary image of Channel 1 (c1) - the FOXP3 immunofluorescence data	~
6	Metadata	T		
G	Groups	rypes		
() (C	Threshold			
0 0	Maskimage			
() C	Savelmage			
1	Savelmages		Select the input impediate (from Nomer And Tuner)	2
() C	Watershed		select the input image c1 v (from NamesAnd types)	
ی 🗇	ConvertObj	ectsTolmage	Name the output image mask_c1	?
() C	GaussianFil	ter		
Image: Contract of the second seco	5 Threshold		Infeshold strategy Global V	1
Image: Contract of the second seco	/ ImageMath		Thresholding method Manual V	?
	KemoveHo Watarahad	es	Manual threshold 0.3	?
0 C	ConvertObi	ectsTolmage		_
0 C	Savelmages	eetsronnage	Threshold smoothing scale 0.0	?
¢ ۲	MeasureOb	jectIntensity		
() C	MeasureOb	jectSizeShape		
ا 🗇	ExportToSp	readsheet		
0.	and Catting	16		
? Ac	diust modules	+ - ^ v		
🕪 Start T	est Mode	Analyze Images	Found 5 rows	_

CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj (D:\John\2020\20200417.3D_LABEL_FREE_SEGMENTATION_PROJECT5\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\		×
File Edit Test Windows Help			
🕑 Images			_
🖾 Metadata	Applies the mask created in the first module to create a background subtracted greyscale image of the FOXP3 immunofluorescence signal	· · · · · · · · · · · · · · · · · · ·	\sim
NamesAndTypes			
🖸 Groups			
🗇 🕑 Threshold			
🗇 🖸 Maskimage			\sim
I Savelmages			
I Savelmages	Select the input image c1 v (from NamesAndTypes)	?	
I Watershed			
	Name the output image Thresh_c1	?	
GaussianFilter	Use objects or an image as a mask? Image	?	1
C Inreshold	integer and integer an		4
	Select image for mask mask_c1 v (from Threshold #05)	?	
Watersbed		2	1
ConvertObjectsTolmage	Invert the mask? O Yes No		1
I MeasureObjectSizeShape			
I ExportToSpreadsheet			
Output Settings View Workspace			
? Adjust modules: + - ^ v			
ID Start Task Mark			
Analyze Images	Found 5 rows		





🔽 Ce	llProfile	r 4.1.3: CP413	_3D_MLN_pipeline.cpproj (I	:\John\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\	- 🗆 ×
File E	dit Te	est Window	s Help		
	8 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8 9 8	Images Metadata NamesAnd Groups	Гуреs	identifies nuclear 'seeds' from the label-free nuclei channel outputted from the UNET	^
	\$ 5	Maskimage			~
	\$ U \$	Savelmages Savelmages		Select the input image c3	?
	\$ U \$	ConvertObj	ectsTolmage	Name the output object Nuclei	?
	\$ 5	Threshold	er	Generate from Distance 🗸	?
	\$ 5	ImageMath RemoveHol	es	Footprint 8	?
	\$ 6 \$ 6	Watershed ConvertObj	ectsTolmage	Downsample 1	?
	\$ 6 \$ 6	Savelmages MeasureOb	jectIntensity		
	\$ 6 \$ 6	MeasureOb ExportToSp	jectSizeShape readsheet		
	Outp	ut Settings	View Workspace		
?	Adj	ust modules:	+ - ^ v		
₽	Start Te	est Mode	Analyze Images	Found 5 rows	

CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cppr	oj (D:\John\2020\2020417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\202110525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\	- 0	×
File Edit Test Windows Help			
C Images C Metadata C NamesAndTypes	Converts the segmented nuclear seeds into an image		^
Groups Groups Groups			
🔿 🗹 Maskimage			\sim
	Select the input objects Nuclei v (from Watershed #09)		- 7
	Name the output image Nuclei_Image		?
	Select the color format uint 16 V		?
🗇 🗹 RemoveHoles			
I MeasureObjectIntensity			
W Export rospreadsneet			
Output Settings View Workspace			
? Adjust modules: + - ^	V I I I I I I I I I I I I I I I I I I I		
Start Test Mode Analyze Images	Count S rouge		
	Found 5 rows		

CellProfiler 4.1.3: CP413 3D MLN pipeline.cpproj ():\John\2020\20200417 3D LAE	EL FREE SEGMENTATION PROJECTS\20210525 biostudies\2 Final 3D MLN example\MATLAB\3D MLN\5 CellProfiler cell measurements\	 ×
File Edit Test Windows Help			
🕑 Images			
🕑 Metadata	Smooths the background/	ther probability map created by the UNET	^
NamesAndTypes			
🖾 Groups			
🗇 🗹 Threshold			
🔿 🗹 Maskimage			 ~
Savelmages			
Saveimages Watershed	Select the input image	c5 v (from NamesAndTypes)	?
ConvertObjectsTolmage	Name the output image	Tissue Mask smoothed	 2
	Nume the output image		
🗇 🗹 Threshold	Sigma	3	?
ImageMath	L		-
I RemoveHoles			
🗇 🗹 Watershed			
🔿 🗹 ConvertObjects Folmage			
Saveimages MassuraObjectIntensity			
MeasureObjectimensity MeasureObjectSizeShape			
Octored Settliness Minutes			
Output Settings View Workspace			
? Adjust modules: + - ^ v			
Start Test Mode Analyze Images	Found 5 rows		

Copyer Steins Very S	CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj (D File Edit Test Windows Help	D:\John\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\ —	
Very Stering Very Stering </th <th>び Images び Metadata び NamesAndTypes び Groups</th> <th>creates a binary image of the smoothed 'background/other' probability map - identifying the region of the image volume occupied by tissue</th> <th>^</th>	び Images び Metadata び NamesAndTypes び Groups	creates a binary image of the smoothed 'background/other' probability map - identifying the region of the image volume occupied by tissue	^
 C Surchrages C Surchrages C Vateshed C Vateshed<td> </td><td></td><td>~</td>	 		~
ConvertSpetting ConvertSpetting Verter Magdath ConvertSpetting ConvertSpetting Verter Magdath ConvertSpetting ConvertSpe	の ビ Savelmages の ビ Savelmages の ビ Watershed	Select the input image Tissue_Mask_smoothed \checkmark (from GaussianFilter #11)	?
Count of line The choid stategy (acbit)		Name the output image TissueMask	?
0 ImageNulfi Threshold smoothing settion 0	 GaussianFilter GuissianFilter	Threshold strategy Global 🗸	?
Output Settings Year Workspace 2 Adjust module:		Thresholding method Minimum Cross-Entropy 🗸	?
© ConvertObject: Torinage Threshold correction factor 10 ? © G MasureObject: NetWorkspee © 10 ? ? © G EportToSpreadheet © 10 ? ? © Log zastings Yiew Workspee ? ? ? Adjuit module: • • • • • •	 ♥ ☑ RemoveHoles ♥ ☑ Watershed 	Threshold smoothing scale 0.0	?
Output Setting: View Workspace Image: Adjust module: Image: Adjust module: Adjust module: Image: Adjust module:		Threshold correction factor 1.0	?
 C S Massure Object Stellsuppe Centre du de la mais de	 	lower and unner hounds on threshold 0.0 10	2
Confunct Setting: View Workspace Adjust module: Adjust module: Adjust module: A v			. 2
Output Settings View Workspace ? Adjust modules • ^		Log transform before thresholding? O Yes No	•
Output Settings View Workspace ? Adjust modules: + -			
	? Adjust modules: + - ^ v		
Start Test Mode Analyze Images Found 5 rows	Start Test Mode Analyze Images	Found 5 rows	

Г

 Images Metadata NamesAndTypes Groups 	Inverts the TissueMask such that it can be used to limit the cell segmentation to the space occupied with cells	
 		
 	Operation Invert V	?
 ♥ ConvertObjectsTolmage ♥ CaussianFilter ♥ Threshold 	Name the output image Inverted_TissueMask	?
	Select the first image TissueMask \checkmark (from Threshold #12)	?
 	Multiply the first image by 1.0	?
SaveImages SaveImages MeasureObjectIntensity	Raise the power of the result by 1.0	?
	Multiply the result by 1.0	?
Export i ospreadsneet	Add to result 0.0	?
	Set values less than 0 equal to 0? Yes O No	?
	Set values greater than 1 equal to 1?	?
	Replace invalid values with 0?	?
	Ignore the image masks? O Yes	?
Output Settings View Workspace		
Adjust modules: + - ^ ,	·	
Start Test Mode	From d From	

CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj	(D:\John\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\	×
File Edit Test Windows Help		
ビ Images ビ Metadata ビ NamesAndTypes ビ Groups	removes small holes from the TissueMask	^
 グ ビ Threshold グ ビ Maskimage ダ ビ Savelmages 		~
 	Select the input image Inverted_TissueMask v (from ImageMath #13)	?
	Name the output image Inverted_TissueMask_deholed	?
 	Size of holes to fill 200	<u> </u>
 		
ConvertObjectsTolmage Savelmages ManuaryObjectIntersity		
VieasureObjectIntensity VieasureObjectSizeShape VieasureObjectSizeShape VieasureObjectSizeShape		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode Analyze Images	Found 5 rows	

CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj (D File Edit Test Windows Help	:\John\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\	
 ☑ Images ☑ Metadata ☑ NamesAndTypes ☑ Groups 	Marker controlled watershed using the previously segmented label-free nuclei as seeds to now find the cell outlines from the label-free actin probability map	^
 Ketadata NamesAndTypes Groups Threshold SaveImages SaveImages SaveImages ConvertObjectsToImage GaussianFilter GaussianFilter SaveImages ConvertObjectsToImage SaveImages MasureObjectsToImage MeasureObjectSizeShape ExportToSpreadsheet 	Select the input image of the processing regimented user included as seen to non-inductive second regiments and included as seen included as seen included as seen included as the second second regiments and included as a second secon	
Output Settings View Workspace ? Adjust modules: + - ^ v Ib Start Test Mode > Analyze Images	Found 5 rows	

Г

CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj (D:\/ohn\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\ —	×
File Edit Test Windows Help		
ビ Images ビ Metadata ビ NamesAndTypes	Converts segmentation results to 'CellMasks'. These are image volumes where each voxel contains the identity of the cell it belongs to	^
Groups		
 		~
Saveimages Saveimages Soviets	Select the input objects Cells v (from Watershed #15)	?
 ♦ Convertobjects romage 	Select the color format uint 16 V	? ?
 ♥ ♥ ImageMath ♥ ♥ RemoveHoles 		
 ✓ Watershed ✓ ConvertObjectsToImage 		
 		
 		
Output Settings View Workspace		
? Adjust modules: + - ^ v		
Start Test Mode Analyze Images	Found 5 rows	



😨 CellProfiler 4.1.3: CP413_3D_MLN_pipeline.cpproj (D:\John\2020\20200417_3D_LABEL_FREE_SEGMENTATION_PROJECTS\20210525_biostudies\2_Final 3D MLN example\MATLAB\3D_MLN\5_CellProfiler_cell_measurements\	– 🗆 X
File Edit Test Windows Help	
ⓒ Images measures cell intensities using the selected images to measure ⓒ MamesAndTypes measures cell intensities using the selected images to measure ⓒ Groups for Ups ⑦ ⓒ Threshold for Ups	^
Image Image	?
	?
Output Settings View Workspace	
? Adjust modules: + - ^ V	
Start Test Mode Analyze Images Found 5 rows	

 Images Metadata NamesAndTypes Groups 	measures cell-object size and sh	ape properties		
Savelinages Savelinages <t< th=""><th>Select object sets to measure</th><th>☑ Cells ☐ Nuclei</th><th>(from Watershed #15) (from Watershed #09)</th><th></th></t<>	Select object sets to measure	☑ Cells ☐ Nuclei	(from Watershed #15) (from Watershed #09)	
ConvertObjects Folmage Savelmages MeasureObjectIntensity	Calculate the Zernike features?	⊖Yes ◉No		
C ExportToSpreadsheet	Calculate the advanced features?	Ves No		
Output Settings View Workspace				

Г

