Open-source deep-learning software for bioimage segmentation

Alice M. Lucas[a,†], Pearl V. Ryder[a,†], Bin Li[b], Beth A. Cimini[a], Kevin W. Eliceiri[b,*], and Anne E. Carpenter[a,*]
[a]Imaging Platform, Broad Institute of MIT and Harvard, Cambridge, MA, 02142; [b]Laboratory for Optical and Computational Instrumentation, Department of Biomedical Engineering, University of Wisconsin at Madison, Madison, WI 53706

ABSTRACT Microscopy images are rich in information about the dynamic relationships among biological structures. However, extracting this complex information can be challenging, especially when biological structures are closely packed, distinguished by texture rather than intensity, and/or low intensity relative to the background. By learning from large amounts of annotated data, deep learning can accomplish several previously intractable bioimage analysis tasks. Until the past few years, however, most deep-learning workflows required significant computational expertise to be applied. Here, we survey several new open-source software tools that aim to make deep-learning–based image segmentation accessible to biologists with limited computational experience. These tools take many different forms, such as web apps, plug-ins for existing imaging analysis software, and preconfigured interactive notebooks and pipelines. In addition to surveying these tools, we overview several challenges that remain in the field. We hope to expand awareness of the powerful deep-learning tools available to biologists for image analysis.

Monitoring Editor
Diane Lidke
University of New Mexico

Received: Dec 1, 2020
Revised: Feb 12, 2021
Accepted: Feb 18, 2021

INTRODUCTION

Microscopy reveals unique aspects of biological specimens, allowing scientists to investigate the relationship between structure and function. Images of biological specimens are rich in data and often difficult to obtain, compelling biologists to extract as much high-quality information from them as feasible. Computational analysis of images can reduce bias by applying a consistent method while minimizing subjectivity and "hands-on" time. Further, these techniques are powerful to detect subtle and unanticipated phenotypes. Open-source software programs such as ImageJ (Schindelin *et al.*, 2015) and CellProfiler (McQuin *et al.*, 2018) allow biologists without formal computational training to apply calculations to images to achieve many different tasks, including identifying individual biological structures—a process known as *segmentation*. However, when biological structures are dimly stained, irregularly shaped, or distinguished from each other predominantly by texture changes rather than intensity value changes, classical segmentation methods are often unsuccessful. Two types of segmentation tasks exist: *semantic* and *instance* segmentation. Semantic segmentation categorizes each pixel into categories but treats multiple objects of the same category as a single entity. Instance segmentation, on the other hand, identifies individual objects as separate entities and is a more common goal in bioimage analysis.

Deep learning promises to accomplish previously intractable bioimage analysis tasks. In classical approaches, a known algorithm is applied to an image in order to accomplish a desired task, using a set of engineered rules. Deep-learning–based methods learn to identify the relevant patterns in the raw input data. This learning is achieved by training complex models, known as deep neural networks, on annotated data sets labeled with the desired output. (Here, we use the terms "models" and "neural networks" interchangeably, although some experts refer to a particular architecture as a network, whereas a model is a trained instance of a given network.) Recent examples of the powerful capabilities of deep-learning models in bioimage analysis include the ability to segment

DOI:10.1091/mbc.E20-10-0660

†These authors contributed equally.

*Address correspondence to: Kevin W. Eliceiri (eliceiri@wisc.edu); Anne E. Carpenter (anne@broadinstitute.org).

Abbreviations used: DIC, differential interference contrast; GPU, graphics processing unit; GUI, graphical user interface; MIB, Microscopy Image Browser; WSI, whole slide image.

© 2021 Lucas, Ryder, *et al*. This article is distributed by The American Society for Cell Biology under license from the author(s). Two months after publication it is available to the public under an Attribution–Noncommercial–Share Alike 3.0 Unported Creative Commons License (http://creativecommons.org/licenses/by-nc-sa/3.0).

"ASCB®," "The American Society for Cell Biology®," and "Molecular Biology of the Cell®" are registered trademarks of The American Society for Cell Biology.

nuclei in fluorescence and brightfield images from many different species with a single model (Caicedo *et al.*, 2019), image restoration by denoising (Krull *et al.*, 2019), and reconstruction of isotropic resolution and subdiffraction structures (Weigert *et al.*, 2018). Technical advancements in the use of deep learning on biological images have been recently reviewed (Moen *et al.*, 2019; Meijering, 2020).

While extremely powerful, until recently, most deep-learning approaches required significant technical expertise to be applied. Over the past few years, an increasing number of deep-learning tools have been released that specifically target users without computational training. The goal of such tools is to make deep learning accessible and time effective for scientists who spend most of their time conducting experiments. Here, we survey currently available open-source software to apply deep learning to bioimage segmentation for users without computational experience. We exclude commercial tools from our survey and instead focus on open-source software with the objective to introduce tools that are accessible to all, favor research reproducibility, and encourage contributions from the bioimaging community. We find four functional groups, described in the following sections and Table 1, and share our perspective on the future of deep learning for biologists.

WEB APPLICATIONS PROVIDE QUICK TESTING OF DEEP-LEARNING METHODS

Several web applications (apps) offer an easy-to-use interface for testing previously trained deep-learning models on a researcher's own microscopy data for segmentation. These web apps include several for light microscopy: StarDist (Schmidt *et al.*, 2018), CellPose (Stringer *et al.*, 2021), NucleAIzer (Hollandi *et al.*, 2020), and DeepCell (Bannon *et al.*, 2018) and one for electron microscopy: CDeep3M-Preview (Haberl *et al.*, 2020) (Table 1). DeepCell also offers object tracking capabilities. Many of these tools are also available in other formats, such as interactive notebooks, locally installable scripts, or cloud-based solutions (Haberl *et al.*, 2018).

A clear advantage of using a web app is the low technical barrier to entry and rapid results, making them good tools to quickly test neural networks. However, several features of currently available web apps limit their utility. First, because the underlying code can change at any time, the user may not be able to analyze the images consistently until these tools enable selecting old versions for analysis. Second, current versions of these web apps are available only for testing pretrained models and have limited preprocessing and configuration options, hindering customization for a particular project's images. Third, large-scale projects may exceed the web app host's computational resources, requiring biologists with hundreds or more images to use another version of the deep-learning tool, such as the locally installed script, or a different software program altogether. Finally, we found that web apps were particularly prone to unpredictable failures and uninformative error messages, which may prevent effective usage.

Two web apps under active development may soon be useful to the cell biology community. Piximi will offer a web-based solution for annotating, training, and evaluating neural networks for phenotype classification and eventually segmentation. ImJoy (Ouyang *et al.*, 2019) provides a web app interface for a multitude of deep-learning–related tasks, although the deep-learning methods for segmentation were under construction at the time of writing. One currently available feature is a web app version of ImageJ that can be used, for example, to apply CellPose.

DEEP-LEARNING PLUG-INS FOR POPULAR IMAGE ANALYSIS TOOLBOXES

Integrating a deep-learning method into a flexible toolbox allows the use of a single, familiar interface for many different tasks. Such toolboxes also allow creation of a streamlined workflow with upstream and downstream steps. Users already familiar with a specific image analysis software platform may therefore want to preferentially learn to use deep-learning plug-ins designed for that software. Several exist for ImageJ. The DeepImageJ plug-in (Gómez-de-Mariscal *et al.*, 2019) provides a framework to test models for several tasks on a researcher's own data; it also provides a user-friendly mechanism to share models. At this time, DeepImageJ provides access to pretrained models, but does not include a mechanism for users to train a model using their data; this can be limiting if existing trained models fail to work well. The CSBDeep toolbox is a collection of ImageJ plug-ins and interactive notebooks (discussed below). At this time, the CSBDeep ImageJ plug-ins can be used to test models for denoising, image restoration, and segmentation. In addition, users can train models for denoising and segmentation with this plug-in, though the developers caution that a computer with a graphics processing unit (GPU) is necessary for timely processing. The interactive notebooks in the CSBDeep toolbox can be additionally used to train all of the models in their collection, a method that allows more customization of the training. These trained models can then be imported into the CSBDeep ImageJ plug-in.

In addition to ImageJ, several other image analysis software packages include plug-ins for deep learning. DeepMIB (Belevich and Jokitalo, 2021) is a deep-learning–based image segmentation plug-in for two- and three-dimensional data sets bundled with the Microscopy Image Browser (MIB), an open-source MATLAB-based image analysis application for light microscopy and electron microscopy. Users can load data sets, test pretrained models or train a model via a graphical user interface (GUI). A pretrained deep-learning model for nucleus segmentation was available as a plug-in for CellProfiler 3 (McQuin *et al.*, 2018) but is not actively maintained in the current CellProfiler 4 release; installation of the necessary TensorFlow libraries made this feature a challenge to use, and the team is turning efforts toward its Piximi app to provide deep-learning capabilities. Finally, ilastik (Berg *et al.*, 2019), an open-source toolkit for interactive machine learning, now includes a beta version for segmentation of images using pretrained deep-learning models. While installing ilastik for use with neural networks is more complex than for typical usage of ilastik, documentation of this process is a work in progress. In addition, the ilastik team is actively developing capability for training neural networks.

Obtaining segmentations from a neural network is usually just one of many steps in image analysis. Biologists may want to compute quantitative metrics on these images, refine segmentations, or use these segmentations as additional input to a new image analysis pipeline. Therefore, the output formats of the tool must be easily readable for further processing and analysis or must directly link to other image processing functions. DeepImageJ and CSBDeep, being integrated as part of ImageJ, provide a major advantage in directly providing access to postprocessing features within ImageJ and ImageJ-integrating tools like Icy (de Chaumont *et al.*, 2012). Furthermore, those who can write code to batch process bioimages using ImageJ can integrate these deep-learning tools into existing workflows. Similarly, the results produced by DeepMIB, such as the trained models and segmentation masks, can be used in other MATLAB scripts for further analysis.

Tool	Website	Available tasks	Bioimage types	For testing or training?	Software format	Usage instructions	Citation
CDeep3M2 Cloud-based image segmentation tasks with pretrained models available for electron micrographs	https://cdeep3m.crbs.ucsd.edu/cdeep3m	Semantic segmentation	• Light • x-ray • Electron microscopy	• Web app: testing for electron micrograph segmentation • Other options: both	• Web app • Google Colab • Docker container app • Singularity container app • AWS cloud	• FAQ • GitHub README and Wiki	Haberl *et al.*, 2018 Haberl *et al.*, 2020
CellPose Nuclear and cytoplasmic segmentation in a web app or local installation with integrated annotation tools for training	http://www.cellpose.org/	Instance segmentation of nuclei and/or cytoplasm	• Fluorescence • Brightfield	• Web app: testing • Local installation: testing and training • Annotation integrated into GUI	• Web app • Local installation with GUI or command line interface • Jupyter Notebook • Google Colab	• Documentation website • GitHub README	Stringer *et al.*, 2020
CSBDeep Toolbox of multiple models in ImageJ plug-ins and Jupyter notebooks	https://csbdeep.bioimagecomputing.com/	• Instance segmentation • Restoration • Combined denoising and instance segmentation	• Fluorescence • Brightfield (H&E) • Electron micrographs	Testing and training	• ImageJ plugins • Jupyter Notebook	• Overview website • Jupyter Notebook text cells • Exercise sheets • Video tutorials	Broaddus *et al.*, 2020 Buchholz *et al.*, 2020 Krull *et al.*, 2019 Schmidt *et al.*, 2018 Weigert *et al.*, 2018
DeepCell Web application for applying already-trained segmentation and tracking models on new images	https://deepcell.org/	• Semantic segmentation • Tracking	• Fluorescence • Phase	Testing	Web app	FAQ	Bannon *et al.*, 2018
DeepImageJ Collection of models within one ImageJ plug-in interface	https://deepimagej.github.io/deepimagej/	• Semantic and instance segmentation • Denoising • Reconstruction • Superresolution • Virtual labeling • Single molecule localization	• Fluorescence • DIC • Phase contrast • Light transmission microscopy	Testing	ImageJ plug-in	• User guide • Video tutorials	Gómez-de-Mariscal *et al.*, 2019

TABLE 1: Overview of deep-learning software for bioimage segmentation.

(Continues)

Tool	Website	Available tasks	Bioimage types	For testing or training?	Software format	Usage instructions	Citation
DeepMIB MATLAB-based tool for segmentation in two or three dimensions	http://mib.helsinki.fi/tutorials_segmentation.html	Semantic segmentation	• Electron microscopy • Brightfield • Fluorescence	Testing and training	• Matlab app • Standalone version (no Matlab needed)	• User guide • Video tutorials	Belevich and Jokitalo, 2021
HistomicsML2 Interactive segmentation for WSIs with integrated active learning	https://github.com/CancerDataScience/HistomicsML2	Semantic segmentation	• WSIs • Histological images • Brightfield	Testing and training	Docker container app	• Documentation website • Video tutorial	Lee *et al.*, 2020
InstantDL Python-based pipeline for classification and segmentation	https://github.com/marrlab/InstantDL	• Semantic segmentation • Instance segmentation • Pixel-wise regression • Two-dimensional classification	• Fluorescence • Brightfield • Medical imaging (CT scan)	Testing and training	• Local installation with command line and config file interface • Docker container app	GitHub README	Waibel *et al.*, 2020
NucleAIzer Nuclear segmentation in a wide variety of image types in a web app or local app containing command line scripts	https://www.nucleaizer.org/	Instance segmentation of nuclei	• Fluorescence • Immunohistochemistry • Hematoxylin & eosin	• Web app: testing • Local installation: testing and training	• Web app • Local installation with command line interface	Explanatory text on web app	Hollandi *et al.*, 2020
ZeroCostDL4Mic Collection of ready-to-use Google Colab notebooks for various image analysis tasks	https://github.com/HenriquesLab/ZeroCostDL4Mic	• Semantic segmentation • Instance segmentation • Denoising • Image-to-image translation • Object detection	• Fluorescence • Electron microscopy	Testing and training	• Google Colab • GPU available through Google	• GitHub Wiki • Colab notebook text cells • Video tutorials	Von Chamier *et al.*, 2020

The described tools are designed for scientists without formal computational training. Definitions: DIC, differential interference contrast; GUI, graphical user interface, a visual method for interacting with software; GPU, graphics processing unit, a specialized type of processor that improves performance, especially for three-dimensional data sets and training models; Jupyter Notebook, interactive notebook to run code interspersed with explanatory text; Google Colab, interactive notebooks hosted in the cloud by Google; WSI, whole slide image.

TABLE 1: Overview of deep-learning software for bioimage segmentation. Continued

INTERACTIVE NOTEBOOKS ARE AN APPROACHABLE INTERFACE FOR TESTING AND TRAINING DEEP-LEARNING MODELS

Interactive notebooks are a common mechanism to share code for deep learning. These notebooks are composed of independent "cells" that can contain code, explanatory text, or GUIs. When interactive notebooks are run, the code cells display output such as numerical data, images, and plots. One type of interactive notebook, the Jupyter Notebook, is typically installed on the user's computer along with the necessary Python language components to run the code (Kluyver *et al.*, 2016). In addition to the CSBDeep ImageJ plug-in described above, the CSBDeep toolbox provides a collection of Jupyter Notebooks for image restoration, denoising, and segmentation. Pretrained models can be tested or users can train models using their own data. Colaboratory (Colab) notebooks are cloud-based interactive notebooks hosted by Google (https://colab.research.google.com/). ZeroCostDL4Mic (Von Chamier *et al.*, 2020) provides a collection of such Colab notebooks, which can be used to train models for a multitude of tasks and image types. The CellPose method for nuclear and cytoplasmic segmentation also includes both Jupyter and Colab notebook options, although the Colab notebook is currently available only for applying trained models (not training new ones).

Interactive notebooks have several advantages for the biologist. As code-based documents, they facilitate reproducibility of an analysis pipeline. Entirely cloud-based interactive notebooks further support reproducibility by providing a sharable environment to run the code. While cloud-based notebooks are typically hosted by an external service such as Google, unlike web apps, users can also download the notebook code and if needed, run the code in a local environment. The interactive nature of these notebooks allows for the graceful integration of crucial validation tasks into the workflow, such as ensuring that the images were correctly loaded into the notebook or visually evaluating the results of a given prediction. A particular advantage of the Colab notebook approach is free access to GPU resources provided by Google, although the free allocation may be insufficient for large-scale experiments (Von Chamier *et al.*, 2020). Although adapting these notebooks typically involves writing code, depending on the extent of the explanatory text amid the code, it is within the capability of an inexperienced researcher willing to learn. Assistance for first-time and infrequent users is critical to increase the accessibility of this type of resource. When assessing which tools to use, we encourage biologists to consider the teaching resources available (Table 1) and note that we have found video tutorials to be particularly helpful for first-time users of bioimage analysis software in general.

PIPELINE-BASED TOOLS COMPRESS DENSE TRAINING AND TESTING CODE INTO EASILY EXECUTABLE SCRIPTS

Local installation of deep-learning tools provides biologists with the opportunity to train and test custom models for their project. CellPose and NucleAIzer both aim to provide a deep-learning model to segment a particular subcellular structure (cells and/or nuclei) in many different image types. HistomicsML2 (Lee *et al.*, 2020) is an application for segmentation on whole slide image (WSI) data sets. The user interfaces differ greatly across these tools. InstantDL (Waibel *et al.*, 2020) and NucleAIzer use the command line, and in some cases a plain-text configuration file, to configure the parameters and run them. CellPose comes with a GUI that includes many helpful preprocessing and postprocessing configuration options, though command line usage is required if the user wishes to perform training or batch testing on their data. The HistomicsML2 software is provided as a Docker container that, once started, can be accessed from the web browser and interacted with through a GUI. In the web-based GUI of HistomicsML2, biologists can annotate their data by dragging and dropping the selected patches into corresponding classes, which are used to train a deep-learning model to segment the image in an active loop. After each training step, regions of high uncertainty are displayed as a heatmap, which can then be annotated and used to provide further training to improve the resulting segmentation.

The tools described in this section typically require the most technical expertise to install and use. Example data, models, and template commands or scripts are often provided as a starting point, but errors may result from the user modifying code in order to adapt to their own data. In such cases, GitHub repositories or the Scientific Community Image Forum (Rueden *et al.*, 2019) are good places to ask for guidance from the project's developers or other users who encountered similar problems. Results can be exported as image masks that can be imported into tools like ImageJ or CellProfiler; alternatively, those already proficient in Python can use InstantDL and CellPose's exported Numpy arrays with Python image analysis libraries for further analysis (Harris *et al.*, 2020). Similarly, the segmentations obtained from a training session in HistomicsML2 are exported as HDF5 files that can be further analyzed from other command line tools.

SUMMARY AND OUTLOOK

Deep learning has revolutionized image analysis tasks in all fields of science, including bioimaging. Open-source software for applying deep learning expands the reach of this powerful approach. Open-source software shares the underlying code, thereby increasing accessibility, promoting transparency (especially for version changes), enabling reproducibility, and encouraging all users to contribute. The software tools highlighted in this Perspective make deep-learning–driven segmentation accessible to users with limited computational experience. Equipped with these tools, biologists can perform previously intractable tasks and uncover new aspects of biology.

We do not here recommend any particular tool for any particular task, simply because each user's data, needs, and computational comfort levels are distinct; each researcher is the best judge of what works for them. As most of these tools provide simple interfaces for applying pretrained models to new data, a user may wish to start by trying many existing pretrained models in order of how comfortable they find each tool to see whether any are satisfactory for their needs. Many image analysis software development teams, including CSBDeep, DeepImageJ, ilastik, ImJoy, and ZeroCostDL4Mic, are actively contributing to the BioImage Model Zoo (bioimage.io), which aims to be a central repository for deep-learning models trained for bioimages. We believe that this initiative will greatly increase access to pretrained models. In general, models trained on a wide variety of image types are more likely to perform well on previously unseen data than those trained on one source (Tzeng *et al.*, 2017), so it may help to look for tools that advertise broad training data for their pretrained models. As tools add more models, and as models become less tailored to specific data sets, we believe that this will become a common and fruitful approach.

If pretrained models are not sufficiently accurate, there are several further considerations to help choose a tool that allows training. One feature to look for is the presence of training mechanisms that prevent overfitting, which occurs when a model becomes too specialized to the training data set and does not generalize well to new data sets. Such overfitting can be prevented with the appropriate use of *regularization* techniques such as *early stopping* (Caruna *et al*., 2001), *dropout* (Srivastava *et al*., 2014), or *data augmentation* methods (Shorten and Khoshgoftaar, 2019). Furthermore, choosing tools that include methods for *transfer learning*, in which knowledge of the model when trained on one task is exploited to improve its learning on a different task, will help increase the resulting prediction accuracy. Software that includes such transfer learning functionalities typically provide multiple pretrained models from which the user can choose as a starting point for fine-tuning on their new task or data set.

Because training configurations may need to be adjusted for different tasks and data sets, another important feature to look for in software is whether it allows for the manipulation of the various training parameters. These parameters include configuring the underlying neural network architecture, how quickly the model learns with each round of training (the learning rate), how many images are used for each learning step (the batch size), and how many times the model should go over the data (epochs). Because each of these settings strongly affects the performance of the model, we encourage biologists to seek software that provides detailed documentation and education on the use and selection of such parameters. If uncertain, we encourage biologists to look for suggestions for default values, because appropriate settings are often roughly in the same range for a given model. We caution that understanding the underlying concepts of deep learning dramatically increases the chances of successful training, so we recommend choosing software that assists sensible decisions through various forms of documentation and education.

In addition to selecting a software for the training itself, researchers will also need to select an image annotation tool to prepare the training images. This choice must be appropriate for both the user's image type and the training software because many annotation tools are restricted to read images of a specific file format or a certain number of channels or dimensionality. Further, many annotation tools enable only a subset of desired annotation types (options include X Y locations, bounding boxes, and full segmentation masks), and it is important to match what your training software expects. Many of the software tools surveyed here include guidance for selecting an annotation tool.

Deep learning allows biologists to accomplish previously intractable image analysis tasks in their research. We hope that the multiple available software tools described here will be useful for those who wish to apply deep learning for segmentation. With better and easier segmentation in hand, biologists will be able to create data sets more quickly and of higher quality, setting the stage for previously impossible discoveries to become possible.

ACKNOWLEDGMENTS
We thank members of the Broad Institute Imaging Platform and the Laboratory for Optical and Computational Instrumentation for thoughtful discussion and feedback on the manuscript. This work was supported by the Center for Open Bioimage Analysis (National Institutes of Health grant 1P41GM135019-01) to A.E.C. and K.W.E.

REFERENCES

Bannon D, Moen E, Schwartz M, Borba E, Kudo T, Greenwald N, Vijayakumar V, Chang B, Pao E, Osterman E, *et al*. (2021). DeepCell Kiosk: scaling deep learning–enabled cellular image analysis with Kubernetes. Nat Methods 18, 43–45.

Belevich I, Jokitalo E (2021). DeepMIB: user-friendly and open-source software for training of deep learning network for biological image segmentation PLOS Comput Biol 17, e1008374.

Berg S, Kutra D, Kroeger T, Straehle CN, Kausler BX, Haubold C, Schiegg M, Ales J, Beier T, Rudy M, *et al*. (2019). Ilastik: interactive machine learning for (bio) image analysis. Nat Methods 16, 1226–1232.

Broaddus C, Krull A, Weigert M, Schmidt U, Myers G (2020). Removing structured noise with self-supervised blind-spot networks. 2020 IEEE 17th Intl Symp Biomed Imaging (ISBI), 159–163.

Buchholz TO, Prakash M, Schmidt D, Krull A, Jug F (2020). DenoiSeg: Joint Denoising and Segmentation. In European Conference on Computer Vision Workshops, Springer, Cham.

Caicedo JC, Goodman A, Karhohs KW, Cimini BA, Ackerman J, Haghighi M, Heng C, Becker T, Doan M, McQuin C, *et al*. (2019). Nucleus segmentation across imaging experiments: the 2018 Data Science Bowl. Nat Methods 16, 1247–1253.

Caruna R, Lawrence S, Giles L (2001). Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping. Adv Neural Inf Process Syst, 381–387.

de Chaumont F, Dallongeville S, Chenouard N, Hervé N, Pop S, Provoost T, Meas-Yedid V, Pankajakshan P, Lecomte T, Le Montagner Y, *et al*. (2012). Icy: an open bioimage informatics platform for extended reproducible research. Nat Methods 9, 690–696.

Gómez-de-Mariscal E, García-López-de-Haro C, Donati L, Unser M, Muñoz-Barrutia A, Sage D (2019). DeepImageJ: a user-friendly plugin to run deep learning models in ImageJ. bioRxiv, 799270.

Haberl MG, Churas C, Tindall L, Boassa D, Phan S, Bushong EA, Madany M, Akay R, Deerinck TJ, Peltier ST, *et al*. (2018). CDeep3M—Plug-and-Play cloud-based deep learning for image segmentation. Nat Methods 15, 677–680

Haberl MG, Wong W, Penticoff S, Je J, Madany M, Borchardt A, Boassa D, Peltier ST, Ellisman MH (2020). CDeep3M-Preview: online segmentation using the deep neural network model zoo. bioRxiv, 010660, https://doi.org/10.1101/2020.03.26.010660.

Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, Wieser E, Taylor J, Berg S, Smith NJ, *et al*. (2020). Array programming with NumPy. Nature 585, 357–362.

Hollandi R, Szkalisity A, Toth T, Tasnadi E, Molnar C, Mathe B, Grexa I, Molnar J, Balind A, Gorbe M, *et al*. (2020). nucleAIzer: a parameter-free deep learning framework for nucleus segmentation using image style transfer. Cell Syst 10, 453–458.e6.

Kluyver T, Ragan-Kelley B, Pérez F, Granger B, Bussonnier M, Frederic J, Kelley K, Hamrick J, Grout J, Corlay S, *et al*. (2016). Jupyter Notebooks—a publishing format for reproducible computational workflows. In: ELPUB, 87–90.

Krull A, Buchholz TO, Jug F (2019). Noise2void—learning denoising from single noisy images. Proc IEEE Conf Comput Vision Pattern Recog, 2129–2137.

Lee S, Amgad M, Mobadersany P, McCormick M, Pollack BP, Elfandy H, Hussein H, Gutman DA, Cooper LA (2020). Interactive classification of whole-slide imaging data for cancer researchers. Cancer Research 81, 1171–1177.

McQuin C, Goodman A, Chernyshev V, Kamentsky L, Cimini BA, Karhohs KW, Doan M, Ding L, Rafelski SM, Thirstrup D, *et al*. (2018). CellProfiler 3.0: next-generation image processing for biology. PLoS Biol 16, e2005970.

Meijering E (2020). A bird's-eye view of deep learning in bioimage analysis. Comput Struct Biotechnol J 18, 2312–2325.

Moen E, Bannon D, Kudo T, Graf W, Covert M, Van Valen D (2019). Deep learning for cellular image analysis. Nat Methods 16, 1233–1246.

Ouyang W, Mueller F, Hjelmare M, Lundberg E, Zimmer C (2019). ImJoy: an open-source computational platform for the deep learning era. Nat Methods 16, 1199–1200.

Rueden CT, Ackerman J, Arena ET, Eglinger J, Cimini BA, Goodman A, Carpenter AE, Eliceiri KW (2019). Scientific Community Image Forum: a discussion forum for scientific image software. PLoS Biol 17, e3000340.

Schindelin J, Rueden CT, Hiner MC, Eliceiri KW (2015). The ImageJ ecosystem: an open platform for biomedical image analysis. Mol Reprod Dev 82, 518–529.

Schmidt U, Weigert M, Broaddus C, Myers G (2018). Cell detection with star-convex polygons. In: Medical Image Computing and Computer Assisted Intervention—MICCAI 2018, Granada, Spain: Springer International Publishing, 265–273.

Shorten C, Khoshgoftaar TM (2019). A survey on image data augmentation for deep learning. J Big Data 6, 60.

Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014). Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15, 1929–1958.

Stringer C, Wang T, Michaelos M, Pachitariu M (2021). Cellpose: a generalist algorithm for cellular segmentation. Nat Methods 18, 100–106.

Tzeng E, Hoffman J, Saenko K, Darrell T (2017). Adversarial discriminative domain adaptation. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

von Chamier L, Jukkala J, Spahn C, Lerche M, Hernández-Pérez S, Mattila PK, Karinou E, Holden S, Can Solak A, Krull A, *et al.* (2020). ZeroCostDL4Mic: an open platform to simplify access and use of deep-learning in microscopy. bioRxiv, 000133. doi: https://doi.org/10.1101/2020.03.20.000133

Waibel DJ, Boushehri SS, Marr C (2020). InstantDL—an easy-to-use deep learning pipeline for image segmentation and classification. BMC Bioinformatics, 22, 1–15.

Weigert M, Schmidt U, Boothe T, Müller A, Dibrov A, Jain A, Wilhelm B, Schmidt D, Broaddus C, Culley S, *et al.* (2018). Content-aware image restoration: pushing the limits of fluorescence microscopy. Nat Methods 15, 1090–1097.