nature methods

BRIEF COMMUNICATION

https://doi.org/10.1038/s41592-018-0261-2

Corrected: Author Correction

U-Net: deep learning for cell counting, detection, and morphometry

Thorsten Falk[1,2,3,20], Dominic Mai[1,2,4,16,20], Robert Bensch[1,2,17,20], Özgün Çiçek[1], Ahmed Abdulkadir[1,5], Yassine Marrakchi[1,2,3], Anton Böhm[1], Jan Deubner[6,7], Zoe Jäckel[6,7], Katharina Seiwald[6], Alexander Dovzhenko[8,18], Olaf Tietz[8,18], Cristina Dal Bosco[8], Sean Walsh[8,18], Deniz Saltukoglu[2,9,10,11], Tuan Leng Tay[7,12,13], Marco Prinz[2,3,12], Klaus Palme[2,8], Matias Simons[2,9,10,14], Ilka Diester[6,7,15], Thomas Brox[1,2,3,7] and Olaf Ronneberger[1,2,19]*

U-Net is a generic deep-learning solution for frequently occurring quantification tasks such as cell detection and shape measurements in biomedical image data. We present an ImageJ plugin that enables non-machine-learning experts to analyze their data with U-Net on either a local computer or a remote server/cloud service. The plugin comes with pretrained models for single-cell segmentation and allows for U-Net to be adapted to new tasks on the basis of a few annotated samples.

Advancements in microscopy and sample preparation techniques have left researchers with large amounts of image data. These data would promise additional insights, more precise analysis and more rigorous statistics, if not for the hurdles of quantification. Images must first be converted into numbers before statistical analysis can be applied. Often, this conversion requires counting thousands of cells with certain markers or drawing the outlines of cells to quantify their shape or the strength of a reporter. Such work is tedious and consequently is often avoided. In neuroscientific studies applying optogenetic tools, for example, quantification of the number of opsin-expressing neurons or the localization of newly developed opsins within the cells is often requested. However, because of the effort required, most studies are published without this information.

Is such quantification not a job that computers can do? Indeed, it is. For decades, computer scientists have developed specialized software that can lift the burden of quantification from researchers in the life sciences. However, each laboratory produces different data and focuses on different aspects of the data for the research question at hand. Thus, new software must be built for each case. Deep learning could change this situation. Because deep learning learns the features relevant for the task from data rather than being hard coded, software engineers do not need to set up specialized software for a certain quantification task. Instead, a generic software package can learn to adapt to the task autonomously from appropriate data—data that researchers in the life sciences can provide by themselves.

Learning-based approaches caught the interest of the biomedical community years ago. Commonly used solutions such as ilastik[1] (http://ilastik.org/) or the trainable WEKA segmentation toolkit[2] (https://imagej.net/Trainable_Weka_Segmentation) enable training of segmentation pipelines by using generic hand-tailored image features. More recently, attention has shifted toward deep learning, which automatically extracts optimal features for the actual image-analysis task, thus obviating the need for feature design by experts in computer science[3–5]. However, widespread use of quantification in the life sciences has been hindered by the lack of generic, easy-to-use software packages. Although the packages Aivia (https://www.drvtechnologies.com/aivia6/) and Cell Profiler[6] (http://cell-profiler.org/) already use deep-learning models, they do not allow for training on new data, thus restricting their application domain to a small range of datasets. In the scope of image restoration, CSBDeep[7] provides an ImgLib2 (https://imagej.net/ImgLib2)-based plugin with models for specific imaging modalities and biological specimens, and allows for integration of externally trained new restoration models. Another approach of bringing deep learning to the life sciences is CDeep3M[8], which ships a set of command-line tools and tutorials for training and applying a residual inception-network architecture for 3D image segmentation. These packages specifically suit researchers with an occasional need for deep learning by providing a cloud-based setup that does not require a local graphics-processing unit (GPU).

The present work provides a generic deep-learning-based software package for cell detection and cell segmentation. For our successful U-Net[3] encoder–decoder network architecture, which has already achieved top ranks in biomedical data-analysis benchmarks[9] and which has been the basis of many deep-learning models in biomedical image analysis, we developed an interface that runs as a plugin in the commonly used ImageJ software[10] (Supplementary Note 1 and Supplementary Software 1–4). In contrast to all previous software packages of its kind, our U-Net can be trained and adapted to

[1]Department of Computer Science, Albert-Ludwigs-University, Freiburg, Germany. [2]BIOSS Centre for Biological Signalling Studies, Freiburg, Germany. [3]CIBSS Centre for Integrative Biological Signalling Studies, Albert-Ludwigs-University, Freiburg, Germany. [4]Life Imaging Center, Center for Biological Systems Analysis, Albert-Ludwigs-University, Freiburg, Germany. [5]University Hospital of Old Age Psychiatry and Psychotherapy, University of Bern, Bern, Switzerland. [6]Optophysiology Lab, Institute of Biology III, Albert-Ludwigs-University, Freiburg, Germany. [7]BrainLinks-BrainTools, Albert-Ludwigs-University, Freiburg, Germany. [8]Institute of Biology II, Albert-Ludwigs-University, Freiburg, Germany. [9]Center for Biological Systems Analysis (ZBSA), Albert-Ludwigs-University, Freiburg, Germany. [10]Renal Division, University Medical Centre, Freiburg, Germany. [11]Spemann Graduate School of Biology and Medicine (SGBM), Albert-Ludwigs-University, Freiburg, Germany. [12]Institute of Neuropathology, University Medical Centre, Freiburg, Germany. [13]Institute of Biology I, Albert-Ludwigs-University, Freiburg, Germany. [14]Paris Descartes University-Sorbonne Paris Cité, Imagine Institute, Paris, France. [15]Bernstein Center Freiburg, Albert-Ludwigs-University, Freiburg, Germany. [16]Present address: SICK AG, Waldkirch, Germany. [17]Present address: ANavS GmbH, München, Germany. [18]Present address: ScreenSYS GmbH, Freiburg, Germany. [19]Present address: DeepMind, London, UK. [20]These authors contributed equally: Thorsten Falk, Dominic Mai, Robert Bensch. *e-mail: ronneber@informatik.uni-freiburg.de

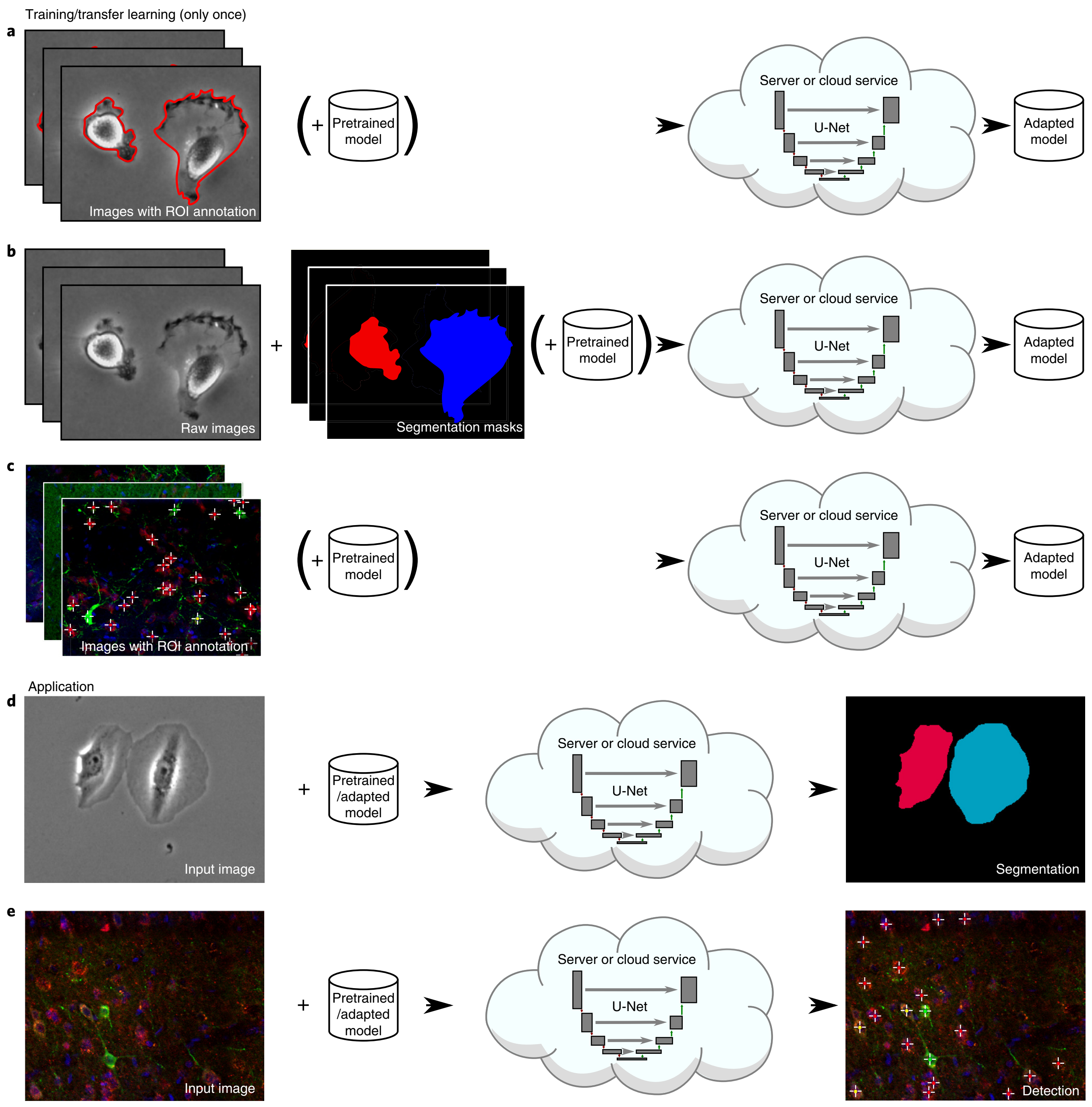

Fig. 1 | Pipelines of the U-Net software. Left to right: input images and model → network training/application (on the local machine, a dedicated remote server or a cloud service) → generated output. **a–c**, Adaptation of U-Net to newly annotated data by using transfer learning. **a**, Segmentation with region of interest (ROI) annotations. **b**, Segmentation with segmentation-mask annotations. **c**, Detection with multipoint annotations. **d**,**e**, Application of the pretrained or adapted U-Net to newly recorded data. **d**, Segmentation. **e**, Detection.

new data and tasks by users themselves, through the familiar ImageJ interface (Fig. 1). This capability enables the application of U-Net to a wide range of tasks and makes it accessible for a wide set of researchers who do not have experience with deep learning. For more experienced users, the plugin offers an interface to adapt aspects of the network architecture and to train on datasets from completely different domains. The software comes with a step-by-step protocol and tutorial showing users how to annotate data for adapting the network and describing typical pitfalls (Supplementary Note 2).

U-Net applies to general pixel-classification tasks in flat images or volumetric image stacks with one or multiple channels. Such tasks include detection and counting of cells, i.e., prediction of a single reference point per cell, and segmentation, i.e., delineation of the outlines of individual cells. These tasks are a superset of the more widespread classification tasks, in which the object of interest is already localized, and only its class label must be inferred. Although adaptation to the detection and segmentation of arbitrary structures in biological tissue is possible given corresponding training data, our experiments focused on cell images, with which the flexibility of U-Net is shown in a wide set of common quantification tasks, including detection in multichannel fluorescence data for dense tissue, and segmentation of cells recorded with different imaging

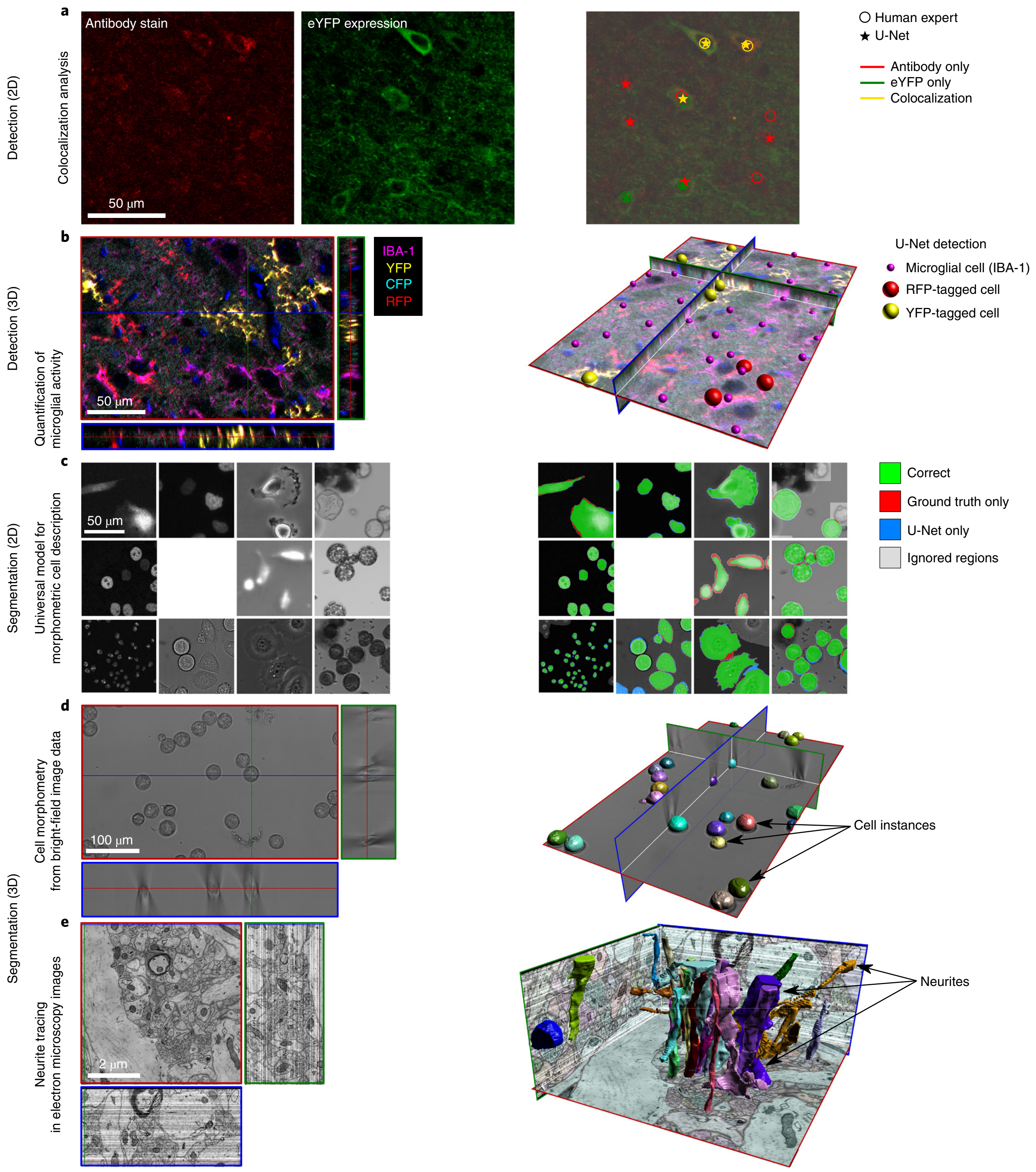

Fig. 2 | Example applications of U-Net for 2D and 3D detection and segmentation. Left, raw data; right, U-Net output (with comparison to human annotation in the 2D cases). **a**, Detection of colocalization in two-channel epifluorescence images. **b**, Detection of fluorescent-protein-tagged microglial cells in five-channel confocal-image stacks. Magenta, all microglia; red, green and cyan: confetti markers; blue, nuclear stain. **c**, Cell segmentation in 2D images from fluorescence, differential interference contrast, phase-contrast and bright-field microscopy by using one joint model. **d**, Cell segmentation in 3D bright-field image stacks. **e**, Neurite segmentation in electron microscopy stacks.

modalities in 2D and 3D (Fig. 2 and Supplementary Note 3). In all cases shown, U-Net would have saved scientists much work. As cross-modality experiments show, the diversity in biological samples is too large for a single software to cover them all out of the box (Supplementary Note 3). Owing to the learning approach, U-Net's applicability is extended from a set of special cases to an unlimited number of experimental settings. The exceptionally precise segmentation results on volumetric bright-field data, whose annotation can push even human experts to their limits, is a particularly strong demonstration of the capabilities of a deep-learning-based automated quantification software (Fig. 2d and Supplementary Note 3).

Critics often stress the need for large amounts of training data to train a deep network. In the case of U-Net, a base network trained on a diverse set of data and our special data-augmentation strategy enable adaptation to new tasks with only one or two annotated images (Methods). Only in particularly difficult cases are more than ten training images needed. Judging whether a model is adequately trained requires evaluation on a held-out validation set and successive addition of training data until no significant improvement on the validation set can be observed (Supplementary Note 3). If computational resources allow, cross-validation with randomly allocated train/test splits avoids selection of a nonrepresentative validation set at the expense of multiple trainings. The manual effort that must be invested to adapt U-Net to a task at hand is typically much smaller than that needed for a minimal statistical analysis of the experimental data. In addition, U-Net offers the possibility to run on much larger sample sizes without requiring any additional effort, thus making it particularly well suited for automated large-scale sample preparation and recording setups, which are likely to become increasingly common in the coming years.

U-Net was optimized for usability in the life sciences. The integration of the software in ImageJ and a step-by-step tutorial make deep learning available to scientists without a computer-science background (Supplementary Videos 1–4). Importantly, the computational load is practicable for common life science laboratory settings (Supplementary Note 1). Adaptation of the network to new cell types or imaging modalities ranges from minutes to a few hours on a single computer with a consumer GPU. If the dedicated compute hardware is not available in the laboratory, common cloud services can be used.

Our experiments provide evidence that U-Net yields results comparable in quality to manual annotation. One special feature of U-Net compared with other automatic annotation tools is the individual influence of the annotator. This feature is advantageous because researchers typically develop individual protocols in which multiple parameters are taken into account without being explicitly mentioned. Owing to their complexity, these labeling rules cannot be reproduced by common automatic labeling tools. However, this advantage can also be a disadvantage: U-Net learns from the provided examples. If the examples are not representative of the actual task, or if the manual annotation in these examples is low quality and inconsistent, U-Net will either fail to train or will reproduce inconsistent annotations on new data. This aspect could also serve as a quality check of the manual annotations. Overall, U-Net cannot correct for low-quality human annotations but is a tool to apply individual labeling rules to large datasets and thereby can save manual annotation effort in a vast variety of quantification tasks.

Online content

Any methods, additional references, Nature Research reporting summaries, source data, statements of data availability and associated accession codes are available at https://doi.org/10.1038/s41592-018-0261-2.

Received: 26 July 2018; Accepted: 19 November 2018;
Published online: 17 December 2018

References

1. Sommer, C, Strähle, C, Koethe, U. & Hamprecht, F. A. in Ilastik: interactive learning and segmentation toolkit in *IEEE Int. Symp. Biomed. Imaging.* 230–233 (IEEE: Piscataway, NJ, USA, 2011).
2. Arganda-Carreras, I. et al. *Bioinformatics* **33**, 2424–2426 (2017).
3. Ronneberger, O., Fischer, P. & Brox, T. U-Net: convolutional networks for biomedical image segmentation. in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015* Vol. 9351, 234–241 (Springer, Cham, Switzerland, 2015).
4. Rusk, N. *Nat. Methods* **13**, 35 (2016).
5. Webb, S. *Nature* **554**, 555–557 (2018).
6. Sadanandan, S. K., Ranefall, P., Le Guyader, S. & Wählby, C. *Sci. Rep.* **7**, 7860 (2017).
7. Weigert, M. et al. *Nat. Methods* https://doi.org/10.1038/s41592-018-0216-7 (2018).
8. Haberl, M. G. et al. *Nat. Methods* **15**, 677–680 (2018).
9. Ulman, V. et al. *Nat. Methods* **14**, 1141–1152 (2017).
10. Schneider, C. A., Rasband, W. S. & Eliceiri, K. W. *Nat. Methods* **9**, 671–675 (2012).

Acknowledgements

This work was supported by the German Federal Ministry for Education and Research (BMBF) through the MICROSYSTEMS project (0316185B) to T.F. and A.D.; the Bernstein Award 2012 (01GQ2301) to I.D.; the Federal Ministry for Economic Affairs and Energy (ZF4184101CR5) to A.B.; the Deutsche Forschungsgemeinschaft (DFG) through the collaborative research center KIDGEM (SFB 1140) to D.M., Ö.Ç., T.F. and O.R., and (SFB 746, INST 39/839,840,841) to K.P.; the Clusters of Excellence BIOSS (EXC 294) to T.F., D.M., R.B., A.A., Y.M., D.S., T.L.T., M.P., K.P., M.S., T.B. and O.R.; BrainLinks-Brain-Tools (EXC 1086) to Z.J., K.S., I.D. and T.B.; grants DI 1908/3-1 to J.D., DI 1908/6-1 to Z.J. and K.S., and DI 1908/7-1 to I.D.; the Swiss National Science Foundation (SNF grant 173880) to A.A.; the ERC Starting grant OptoMotorPath (338041) to I.D.; and the FENS-Kavli Network of Excellence (FKNE) to I.D. We thank F. Prósper, E. Bártová, V. Ulman, D. Svoboda, G. van Cappellen, S. Kumar, T. Becker and the Mitocheck consortium for providing a rich diversity of datasets through the ISBI segmentation challenge. We thank P. Fischer for manual image annotations. We thank S. Wrobel for tobacco microspore preparation.

Author contributions

T.F., D.M., R.B., Y.M., Ö.Ç., T.B. and O.R. selected and designed the computational experiments. T.F., R.B., D.M., Y.M., A.B. and Ö.Ç. performed the experiments: R.B., D.M., Y.M. and A.B. (2D), and T.F. and Ö.Ç. (3D). R.B., Ö.Ç., A.A., T.F. and O.R. implemented the U-Net extensions into caffe. T.F. designed and implemented the Fiji plugin. D.S. and M.S. selected, prepared and recorded the keratinocyte dataset PC3-HKPV. T.F. and O.R. prepared the airborne-pollen dataset BF1-POL. A.D., S.W., O.T., C.D.B. and K.P. selected, prepared and recorded the protoplast and microspore datasets BF2-PPL and BF3-MiSp. T.L.T. and M.P. prepared, recorded and annotated the data for the microglial proliferation experiment. J.D., K.S. and Z.J. selected, prepared and recorded the optogenetic dataset. I.D., J.D. and Z.J. manually annotated the optogenetic dataset. I.D., T.F., D.M., R.B., Ö.Ç., T.B. and O.R. wrote the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41592-018-0261-2.

Reprints and permissions information is available at www.nature.com/reprints.

Correspondence and requests for materials should be addressed to O.R.

Publisher's note: Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Methods

Network architecture. U-Net is an encoder–decoder-style neural network that solves semantic segmentation tasks end to end; it extends the fully convolutional network from Long et al.[11] (Supplementary Fig. 1). Its 26 hidden layers can be logically grouped in two parts: (i) an encoder that takes an image tile as input and successively computes feature maps at multiple scales and abstraction levels thus yielding a multilevel, multiresolution feature representation, and (ii) a decoder that takes the feature representation and classifies all pixels/voxels at original image resolution in parallel. Layers in the decoder gradually synthesize the segmentation, starting at low-resolution feature maps (describing large-scale structures) and moving to full-resolution feature maps (describing fine-scale structures).

The encoder is a vgg-style convolutional neural network[12]. It consists of the repeated application of two convolutions (without padding), each followed by a leaky rectified linear unit (ReLU) with a leakage factor of 0.1 and a max-pooling operation with stride two halving the resolution of the resulting feature map. Convolutions directly following down-sampling steps double the number of feature channels.

The decoder consists of repeated application of an up-convolution (an up-convolution is equivalent to a bed-of-nails up-sampling by a factor of two followed by a convolution with an edge length of two) that halves the number of feature channels, then concatenation with the cropped encoder feature map at corresponding resolution and two convolutions, each followed by leaky ReLUs with a leakage factor of 0.1.

At the final layer, a 1×1 convolution is used to map feature vectors to the desired number of classes K.

For training, the final K-class scores are transformed to pseudoprobabilities with soft-max before they are compared to the ground-truth annotations in one-hot encoding using cross-entropy.

If not explicitly mentioned, operations are isotropic, and convolutions use a kernel edge length of three pixels. We use only the valid part of each convolution; i.e., the segmentation map contains only the pixels for which the full context is available in the input image tile.

This fully convolutional network architecture allows users to freely choose the input-tile shape, with the restriction that the edge lengths of feature maps before max-pooling operations must be even. Thus, arbitrarily large images can be processed by using an overlap-tile strategy in which the stride is given by the output-tile shape. To predict pixels in the border region of an image, missing context is extrapolated by mirroring the input image.

The ImageJ plugin presents only valid tile shapes to the user. For 2D networks, the user can also let the plugin automatically choose an optimal tiling according to the available GPU memory. For this process, the amount of memory used by the given architecture for different tile sizes must be measured on the client PC beforehand, and corresponding information must be stored to the model definition file by using the provided MATLAB script caffe-unet/matlab/unet/measureGPUMem.m.

3D U-Net. The 3D U-Net architecture for volumetric inputs includes only a slight modification of its 2D counterpart. First, input tiles have a shape of $236 \times 236 \times 100$ voxels. Second, owing to memory limitations, we halve the number of output features of each convolution layer except for the last up-convolution step, which produces only 32 channels. Third, to address anisotropic image voxel sizes, convolutions and pooling at the finest-resolution levels are 2D until the voxel extents are approximately isotropic. At all remaining resolution levels, operations are applied along all three dimensions.

Weighted soft-max cross-entropy loss. We use pixel-weighted soft-max cross-entropy loss to enable changing the influence of imbalanced classes in semantic segmentation. The loss is computed as

$$l(I) := -\sum_{x\in\Omega} w(\boldsymbol{x}) \log \frac{\exp(\hat{y}_{y(x)}(\boldsymbol{x}))}{\sum_{k=0}^{K} \exp(\hat{y}_k(\boldsymbol{x}))}$$

where x is a pixel in image domain Ω, $\hat{y}_k : \Omega \to \mathbb{R}$ is the predicted score for class $k\in\{0,\ldots,K\}$, K is the number of classes, and $y{:}\Omega\to\{0,\ldots,K\}$ is the ground-truth segmentation map. Thus, $\hat{y}_{y(x)}(\boldsymbol{x})$ is the predicted score for ground-truth class $y(\boldsymbol{x})$ at position $\boldsymbol{x}$. As defined above, $w : \Omega \to \mathbb{R}_{\geq 0}$ is the pixelwise-loss-weight map.

We use loss weighting to optimize class balancing and handle regions without annotations (termed 'ignored regions' below), by using weight map w_{bal}. We additionally enforce instance separation, but using weight w_{sep}, as described in the following sections. The final loss weights are then given by

$$w := w_{\text{bal}} + \lambda w_{\text{sep}}$$

where $\lambda\in R_{\geq 0}$ controls the importance of instance separation.

Class balancing and regions with unknown ground truth. In our images, most pixels belong to the background class, and in many cases, this class is homogeneous and easy to learn. Therefore, we decrease the weight of background pixels by the factor $v_{\text{bal}}\in[0,1]$, as compared with foreground pixels, thus resulting in the class-balancing weight function

$$w_{bal}(\boldsymbol{x}) := \begin{cases} 1 & y(\boldsymbol{x}) > 0 \\ v_{\text{bal}} & y(\boldsymbol{x}) = 0 \\ 0 & y(\boldsymbol{x})\,\text{unknown (i.e., ignored regions)} \end{cases}$$

We observed slightly better segmentation results when replacing the step-shaped cutoff at the edges of foreground objects by a smoothly decreasing Gaussian function for the weighted loss computation; therefore, we define

$$w'_{bal}(\boldsymbol{x}) := \begin{cases} 1 & y(\boldsymbol{x}) > 0 \\ v_{\text{bal}} + (1-v_{bal}) \cdot \exp\left(-\dfrac{d_1^2(\boldsymbol{x})}{2\sigma_{\text{bal}}^2}\right) & y(\boldsymbol{x}) = 0 \\ 0 & y(\boldsymbol{x})\,\text{unknown (i.e., ignored regions)} \end{cases}$$

where $d_1(\boldsymbol{x})$ is the distance to the closest foreground object, and σ_{bal} is the s.d. of the Gaussian.

Detection. The detection tasks are reformulated as segmentation tasks by rendering a segmentation map containing a small disk (2D) or a small ball (3D) at each annotated location. Each disk or ball is surrounded by a ring-shaped or spherical-shell-shaped ignored region with outer radius r_{ign}, in which both w_{bal} and w'_{bal} are set to zero. These regions help stabilize the detection process in case of inaccurate annotations but are not essential to the approach.

Instance segmentation. Semantic segmentation classifies the pixels/voxels of an image. Therefore, touching objects of the same class will end up in one joint segment. Most often, users want to measure the properties of individual object instances (for example, cells). Therefore, the pure semantic segmentation must be turned into an instance-aware semantic segmentation. To do so, we insert an artificial 1-pixel-wide background ridge between touching instances in the ground-truth segmentation mask (Supplementary Fig. 2a). To force the network to learn these ridges, we increase their weight in the loss computation, such that the thinnest ridges have the highest weights. We approximate the ridge width at each pixel by the sum of the distance d_1 to its nearest instance and the distance d_2 to its second-nearest instance. From this, we compute the weight map as

$$w_{\text{sep}}(\boldsymbol{x}) := \exp\left(-\frac{(d_1(\boldsymbol{x}) + d_2(\boldsymbol{x}))^2}{2\sigma_{\text{sep}}^2}\right)$$

which decreases following a Gaussian curve with s.d. σ_{sep} (Supplementary Fig. 2b).

Tile sampling and augmentation. Data augmentation is essential to teach the expected appearance variation to the neural network with only a few annotated images. To become robust to translations and to focus on relevant regions in the images, we first randomly draw spatial locations of image tiles that are presented to the network from a user-defined probability density function. For all our experiments, we used the normalized weight map w_{bal} to sample the spatial location; i.e., foreground objects are presented to the network ten times more often (according to our selection of $v_{\text{bal}} = 0.1$) than background regions. Tiles centered around an ignored region are never selected during training. We then draw a random rotation angle (around the optical axis in 3D) from a uniform distribution in a user-defined range. Finally, we generate a smooth deformation field by placing random displacement vectors with user-defined s.d. of the magnitude for each component on a very coarse grid. These displacement vectors are used to generate a dense full-resolution deformation field by using bicubic interpolation (Supplementary Fig. 3). Rigid transformations and elastic deformations are concatenated to look-up intensities in the original image during tile generation.

We additionally apply a smooth strictly increasing intensity transformation to become robust to brightness and contrast changes. The intensity mapping curve is generated from a user-defined number of equidistant control points in the normalized [0,1] source intensity range. Target intensities at the control points are drawn from uniform distributions with user-defined ranges. The sampling process enforces increased intensities at subsequent control points, and spline interpolation between the control points ensures smoothness. All data augmentation is applied on the fly to the input image, label map and weight map during network training[13].

Training. All networks were trained on an nVidia TITAN X with 12 GB GDDR5 RAM, by using cuda 8 and cuDNN 6 with caffe[14] after our proposed extensions were applied. The initial network parameters were drawn from a Gaussian distribution with s.d. $\sigma = \sqrt{2/n_{\text{in}}}$, where n_{in} is the number of inputs of one neuron of the respective layer[15]. For all experiments, the raw image intensities per channel were normalized to the [0,1] range before training, by using

$$\hat{\mathrm{I}} := \frac{I - \min\{I\}}{\max\{I\} - \min\{I\}}$$

where I is the raw intensity, and $\hat{\mathrm{I}}$ is the normalized intensity.

Transfer learning. Adaptation of a pretrained U-Net to a new dataset by using annotated data is called transfer learning or fine-tuning. Transfer learning leverages the knowledge about the different cell datasets already learned by the U-Net and usually requires considerably fewer annotated data and training iterations than training from scratch.

Transfer to a new dataset is based on the same training protocol as described above. The user must provide only raw images and corresponding annotations as ImageJ ROIs or pixel masks (detection, one multipoint ROI per class and image; segmentation, one regional ROI per object or pixel masks) (Supplementary Note 2). The plugin performs image rescaling, pixel-weight $w(\boldsymbol{x})$ generation and parametrization of the caffe training software.

Evaluation metrics. *Object intersection over union.* The intersection over union (IoU) measures how well a predicted segmentation matches the corresponding ground-truth annotation by dividing the intersection of two segments by their union. Let $\mathcal{S} := \{s_1, \ldots, s_N\}$ be a set of N pixels. Let $\mathcal{G} \subset \mathcal{S}$ be the set of pixels belonging to a ground-truth segment and $\mathcal{P} \subset \mathcal{S}$ be the set of pixels belonging to the corresponding predicted segment. The IoU is defined as

$$M_{\mathrm{IoU}}(\mathcal{G}, \mathcal{P}) := \frac{|\mathcal{G} \cap \mathcal{P}|}{|\mathcal{G} \cup \mathcal{P}|}$$

IoU is a widely used measure, for example, in the Pascal VOC challenge[16] or the ISBI cell-tracking challenge[17]. $M_{\mathrm{IoU}} \in [0,1]$, with 0 meaning no overlap and 1 meaning a perfect match. In our experiments, a value of ~0.7 indicates a good segmentation result, and a value of ~0.9 is close to human annotation accuracy.

We first determine the predicted objects by performing connected component labeling with eight-neighborhood on the binary output of the U-Net. This procedure yields candidate segments $\mathcal{P}_i$. Then we compute the IoU for every pair of output and ground-truth segments $\mathcal{G}_j: M_{\mathrm{IoU}}(\mathcal{G}_j, \mathcal{P}_i)$ and apply the Hungarian algorithm on M_{IoU} to obtain 1:1 correspondences maximizing the average IoU. Unmatched segments or matches with zero IoU are considered false positives and false negatives, respectively. The average IoU is computed on the basis of the ground-truth annotations; i.e., false negatives contribute to the average IoU with a value of 0. False positives, however, are not considered in the average IoU. We provide a detailed analysis of false-positive segmentations produced by U-Net (Supplementary Note 3).

F measure. We measure the quality of a detector by using the balanced F_1 score (denoted F measure), which is the harmonic mean of precision and recall. Let G be the set of true object locations and P be the set of predicted object locations. Then precision and recall are defined as

$$M_{\mathrm{Precision}} := \frac{|G \cap P|}{|P|} \quad \text{and} \quad M_{\mathrm{Recall}} := \frac{|G \cap P|}{|G|}.$$

The F measure is then given by

$$F_1 : 2 \cdot \frac{M_{\mathrm{Precision}} \cdot M_{\mathrm{Recall}}}{M_{\mathrm{Precision}} + M_{\mathrm{Recall}}}$$

Pixel-accurate object localization is nearly impossible to reach and is rarely necessary in biomedical applications; therefore, we introduce a tolerance d_{match} for matching detections and annotations. To obtain one-to-one correspondences, we first compute the pairwise distances of ground-truth positions to detections. Predictions with distance greater than d_{match} to any ground-truth position can be directly classified as false positives. Similarly, ground-truth positions without detections in the d_{match} range are classified as false negatives. We apply the Hungarian algorithm to the remaining points to obtain an assignment of predictions to ground-truth positions minimizing the total sum of distances of matched positions. Correspondences with distance greater than d_{match} and unmatched positions in either ground-truth or prediction are treated as false-negative or false-positive detections as before.

Statistics. In the microglial detection experiment, significant differences between random clone distribution and clone clustering are reported. We compared the actual measurements obtained through U-Net detection with a Monte Carlo experiment simulating the null hypothesis of equally distributed confetti-marked microglia. We report the ninety-eighth-percentile range of 10,000 simulation runs (bootstrap iterations) leading to a two-tailed nonparametric test. If the measurement is outside the ninety-eighth-percentile, the null hypothesis does not hold, with a P value <0.02.

Code availability. We have provided prebuilt binary versions of the U-Net caffe extensions for Ubuntu Linux 16.04 at https://lmb.informatik.uni-freiburg.de/resources/opensource/unet/. Our changes to the source code of the publicly available caffe deep-learning framework[14] are additionally provided (https://github.com/BVLC/caffe/) as a patch file with detailed instructions on how to apply the patch and build our caffe variant from the source on the project website and within the plugin's Help function. Binary installation requires only unpacking the archive and installing the required third-party libraries, as can be done within several minutes on an Ubuntu 16.04 machine, depending on the internet connection for fetching the packages. Building from scratch requires installing the dependent development libraries, checking out the given tagged version of the BVLC caffe master branch and applying the patch. With a normal internet connection, package installation requires a few minutes. Cloning the BVLC master repository requires less than a minute, and applying the patch imposes no measurable overhead. Configuring and building the package requires approximately 10 to 15 min. The U-Net segmentation plugin for Fiji/ImageJ is available at http://sites.imagej.net/Falk/plugins/ or through the ImageJ updater within Fiji. Source code is included in the plugin jar File Unet_Segmentation.jar. Installation with the Fiji updater requires only a few seconds. The trained caffe-models for the 2D and 3D U-Net are available at https://lmb.informatik.uni-freiburg.de/resources/opensource/unet/.

Reporting Summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

Datasets F1-MSC, F2-GOWT1, F3-SIM, F4-HeLa, DIC1-HeLa, PC1-U373 and PC2-PSC are from the ISBI Cell Tracking Challenge 2015 (ref. [17]). Information on how to obtain the data can be found at http://celltrackingchallenge.net/datasets.html, and free registration for the challenge is currently required. Datasets PC3-HKPV, BF1-POL, BF2-PPL and BF3-MiSp are custom and are available from the corresponding author upon reasonable request. Datasets for the detection experiments partially contain unpublished sample-preparation protocols and are currently not freely available. After protocol publication, datasets will be made available on an as-requested basis. Details on sample preparation for our life science experiments can be found in Supplementary Note 3 and the Life Sciences Reporting Summary.

References

11. Long, J., Shelhamer, E. & Darrell, T. Fully convolutional networks for semantic segmentation. in *IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)* 3431–3440 (IEEE, Piscataway, NJ, USA, 2015).
12. Simonyan, K. & Zisserman, A. Preprint at https://arxiv.org/abs/1409.1556 (2014)
13. Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T. & Ronneberger, O. 3D U-Net: learning dense volumetric segmentation from sparse annotation. in *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2016* Vol. 9901, 424–432 (Springer, Cham, Switzerland, 2016).
14. Jia, Y. et al. Preprint at https://arxiv.org/abs/1408.5093 (2014).
15. He, K., Zhang, X., Ren, S. & Sun, J. Preprint at https://arxiv.org/abs/1502.01852 (2015).
16. Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J. & Zisserman, A. *Int. J. Comput. Vis.* **88**, 303–338 (2010).
17. Maška, M. et al. *Bioinformatics* **30**, 1609–1617 (2014).

nature research

Corresponding author(s): Olaf Ronneberger

Reporting Summary

Statistical parameters

When statistical analyses are reported, confirm that the following items are present in the relevant location (e.g. figure legend, table legend, main text, or Methods section).

n/a	Confirmed	
☐	☒	The exact sample size (n) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒	An indication of whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistics including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. F, t, r) with confidence intervals, effect sizes, degrees of freedom and P value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's d, Pearson's r), indicating how they were calculated
☒	☐	Clearly defined error bars *State explicitly what error bars represent (e.g. SD, SE, CI)*

Our web collection on statistics for biologists may be useful.

Software and code

Policy information about availability of computer code

Data collection	Optogenetics experiment (2D detection): ZEN2010 B SP1 Version 6.0.0.485; Microglia experiment (3D detection): Olympus FluoView 1000; Cell segmentation: BF2-PPL,. BF3-MiSp: LA (Live Acquisition) Software, TILL Photonics GmbH, Germany, Version 2.2.1.0, PC3-HKPV: ZEISS AxioVision All other datasets are thirdparty.
Data analysis	caffe_unet (custom, binaries and source code provided as Supplementary Software 1-4). More recent versions can be found at https://lmb.informatik.uni-freiburg.de/resources/opensource/unet/ caffe_unet is a variant of the open source deep learning framework caffe (https://github.com/BVLC/caffe). It is the core of this study. Our U-Net segmentation plugin for Fiji (ImageJ) is a front-end that interfaces caffe from ImageJ to provide easy integration of segmentation and detection using deep learning into Fiji protocols for non-computer scientists. U-Net Segmentation plugin (custom, provided via the Fiji community repository sites.imagej.net): version at time of submission: https://sites.imagej.net/Falk/plugins/Unet_Segmentation.jar-20181112152803 The plugin is maintained to work with the most recent Fiji version and requires Java 8. Future updates may break backwards-compatibility if the structure of the ImageJ API changes. Fiji (ImageJ) (https://fiji.sc). Version at time of submission: 1.52h. Fiji is a variant of the open source image processing software ImageJ with many useful extensions for image analysis in the life sciences.

MATLAB (https://de.mathworks.com/products/matlab.html): version used in this study: R2016a
MATLAB is a commercial general purpose suite optimized for fast and versatile numeric operations on multi-dimensional data. It provides a wide range of toolboxes for various tasks, including image processing. Many of the experiments for this study were performed with MATLAB as frontend, but it is not required to reproduce any of our findings.

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors/reviewers upon request. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Research guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A list of figures that have associated raw data
- A description of any restrictions on data availability

The datasets F1-MSC, F2-GOWT1, F3-SIM, F4-HeLa, DIC1-HeLa, PC1-U373, and PC2-PSC are from the ISBI Cell Tracking Challenge 2015. Informations on how to obtain the data can be found at http://celltrackingchallenge.net/datasets.html and at time of publication required free-of-charge registration for the challenge.

The datasets PC3-HKPV, BF1-POL, BF2-PPL, and BF3-MiSp are custom and are available from the corresponding author upon reasonable request.

Datasets for the detection experiments partially contain unpublished sample preparation protocols, and are currently not freely available. Upon protocol publication datasets will be made available on request-basis.

Field-specific reporting

Please select the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

[✓ Life sciences] ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/authors/policies/ReportingSummary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	No statistical methods were used to predetermine sample sizes in the microglia experiment. We ensured that they were similar to those generally employed in the field. We chose a random 25% training / 75% test data split to provide a sufficient amount of training data to the network while still having enough data in the test set to draw statistically significant conclusions. Further experiments with other splits were not performed due to limited computational resources.
Data exclusions	All available annotated images were included without selection. The subset of raw images for annotation was randomly chosen from all available images and randomly split into training and test sets. A validation set was not required, because we did not optimize hyperparameters of our model besides the number of training iterations, which we selected so that we obtained the best average results on the test set, but the full curves are given for reference and show that the number of iterations is not a critical parameter.
Replication	Facial nerve transection (microglia experiment) was performed in at least three separate occasions and cohorts were randomly assigned to all test groups (i.e., time after injury). All replication attempts were successful. We repeated the finetuning experiments for 2D cell segmentation on randomly chosen subsets of the training data. If the training set is sufficiently large (100 cells or more) the prediction variance tends towards zero showing high reproducibility.
Randomization	Sample allocation was random in all stages of the experiments. A coworker of TLT randomly assigned numbers to mice, slides and images for data acquisition and processing (microglia experiment).
Blinding	Only in the microglia experiment different groups were analyzed and compared. The experiment was performed in a blinded manner due to random assignment of numbers to mice, slides and images for data acquisition and processing. Final results were reassigned to their respective test groups by TLT and DM.

Reporting for specific materials, systems and methods

Materials & experimental systems

n/a	Involved in the study
☒	☐ Unique biological materials
☐	☒ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology
☐	☒ Animals and other organisms
☒	☐ Human research participants

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Antibodies

Antibodies used

Optogenetics experiment: We used antibodies against NeuN 1:500 (EMD Millipore Corp., Catalog Nr.: ABN78, polyclonal produced in rabbit, Lot: 2951836), CaMKIIα (A-1, Santa Cruz Biotechnology, Catalog Nr.: Sc-13141, monoclonal produced in mouse, LOT: B1712) and Parvalbumin (Sigma-Aldrich, Catalog Nr.: P3088, clone PARV-19, monoclonal produced in mouse, LOT: 122M4774V) as primary antibodies. As secondary antibodies we used Goat Anti-Rabbit IgG (H+L), Alexa Fluor® 647 conjugate (Sigma-Aldrich, Catalog Nr.: AP187SA6, polyclonal, LOT: 2277868) and Donkey Anti-Mouse IgG, Alexa Fluor® 647 conjugate (Sigma-Aldrich, Catalog Nr.: AP192SA6, polyclonal, LOT: 2613129).

Microglia experiment: 1:500 rabbit anti-Iba-1 (Wako 019-19741); 1:1,000 donkey anti-rabbit IgG (H+L) Highly Cross-Adsorbed Secondary Antibody, Alexa Fluor 647 (Life Technologies A-31573)

Validation

Optogenetics experiment: Validation of the antibodies was done by immunoelectrophoresis and/or ELISA by the manufacturer to ensure specificity and minimal cross-reaction with rat serum proteins and/or extensive use in various publications.

NeuN:
Immunocytochemistry Analysis: A 1:500 dilution from a representative lot detected NeuN in cultured embryonic E18 rat hippocampus neurons. [EMD Millipore Corp.]
Immunohistochemistry Analysis: A 1:500 dilution from a representative lot detected NeuN-positive neurons in human (cerebellum and cerebral cortex) and mouse (hippocampus) brain tissue sections. [EMD Millipore Corp.]
A representative lots immunostained 4% paraformaldehyde-fixed neurons isolated from snail brain (Nikolić, L., et al. (2013). J. Exp. Biol. 216(Pt 18):3531-3541).
Representative lots detected NeuN-positive neurons in frozen brain sections from 4% paraformaldehyde-perfused mice (Cherry, J.D., et al. (2015). J. Neuroinflammation. 12:203; Huang, B., et al. (2015). Neuron. 85(6):1212-1226; Ataka, K., et al. (2013). PLoS One. 8(11):e81744).
A representative lots detected NeuN-positive neurons in formalin-fixed and paraffin-embedded amyotrophic lateral sclerosis (ALS) human brain tissue sections (Fratta, P., et al. (2015). Neurobiol. Aging. 36(1):546.e1-7).
A representative lot detected NeuN-positive neurons in rat brain tissue sections (Mendonça, M.C., et al. (2013).Toxins. 5(12):2572-2588).
Representative lot detected NeuN in mouse and rat brain tissue lysates (Huang, B., et al. (2015). Neuron. 85(6):1212-1226; Mendonça, M.C., et al. (2013).Toxins. 5(12):2572-2588).

CamKII:
Maurya, SK. et al. 2018. Cell Rep. 24: 2919-2931. ; Lee, Y. et al. 2018. Mol. Neurobiol. 55: 5658-5671. ; Go, J. et al. 2018. Int. J. Mol. Med. 42: 1875-1884. ; Oliver, RJ. et al. 2017. Genes Brain Behav. ; Zhang, Y. et al. 2017. J. Neurosci. 37: 9741-9758. ; Raffeiner, P. et al. 2017. Oncotarget. 8: 3327-3343. ; Yan, X. et al. 2016. PLoS ONE. 11: e0162784. ; Deng, H. et al. 2016. J. Neurosci. 36: 7580-8. ; Volle, J. et al. 2016. Cell Rep. 15: 2400-10. ; Kim, H. et al. 2016. Mol. Neurobiol. ; Sheng, L. et al. 2015. J. Neurosci. 35: 1739-52. ; Cipolletta, E. et al. 2015. PloS one. 10: e0130477. ; Gratuze, M. et al. 2015. Hum. Mol. Genet. 24: 86-99. ; Wang, S. et al. 2015. J Biomed Res. 29: 370-9. ; Okada, R. et al. 2015. Neuroscience. 299: 134-45. ; Okada, R. et al. 2014. Neurochem. Res. 39: 875-82. ; Mizuki, D. et al. 2014. J Ethnopharmacol. 156: 16-25. ; Mizuki, D. et al. 2014. J. Pharmacol. Sci. 124: 457-67. ; Kim, H. et al. 2014. Mol Brain. 7: 77. ; Le, XT. et al. 2013. Neurochem. Res. 38: 2201-15. ; Cao, C. et al. 2013. PLoS biology. 11: e1001478. ; Yen, YH. et al. 2011. Mol Cell Proteomics. -. ; Zhao, Q. et al. 2010. J Ethnopharmacol. 131: 377-385. ; España, J. et al. 2010. Biol. Psychiatry. 67: 513-521. ; Zhang, G.R. et al. 2009. Neuroscience. 159: 308-315. ; Cheng, HH. et al. 2009. J. Neurosci. Res. 87: 460-469. ; Chang, CW. et al. 2007. Proteomics Clin Appl. 1: 1499-1512. ; Saha, S. et al. 2007. Brain Res. 1148: 38-42. ; Saha, S. et al. 2006. Biochem. Biophys. Res. Commun. 350: 444-449.

PV:
Recognizes parvalbumin in a Ca2+ ion-dependent manner. Does not react with other members of the EF-hand family such as calmodulin, intestinal calcium-binding protein, S100A2 (S100L), S100A6 (calcyclin), the α chain of S-100 (i.e. in S-100a and S-100ao), or the β chain (i.e. in S-100a and S-100b). [Sigma-Aldrich]

Goat anti rabbit IgG Alexa Fluor 647:
Based on immunoelectrophoresis and/or ELISA, the antibody reacts with whole molecule rabbit IgG. It also reacts with the light chains of other rabbit immunoglobulins. No antibody was detected against non-immunoglobulin serum proteins.
The antibody has been tested by ELISA and/or solid-phase absorbed to ensure minimal cross-reaction with human, mouse, and rat serum proteins, but may cross-react with immunoglobulins from other species. [Sigma-Aldrich]

Donkey anti mouse IgG Alexa Fluor 647:
Based on immunoelectrophoresis and/or ELISA, the antibody reacts with whole molecule mouse IgG. It also reacts with the light chains of other mouse immunoglobulins. No antibody was detected against non-immunoglobulin serum proteins.

The antibody has been tested by ELISA and/or solid-phase absorbed to ensure minimal cross-reaction with bovine, chicken, goat, guinea pig, Syrian hamster, horse, human, rabbit, rat, and sheep serum proteins, but may cross-react with immunoglobulins from other species. [Sigma-Aldrich]

Microglia experiment: As provided on manufacturer's website (see also [23] Tay, T. L. et al. Neuroscience 20, 793–803 (2017))

Animals and other organisms

Policy information about studies involving animals; ARRIVE guidelines recommended for reporting animal research

Laboratory animals	Optogenetics experiment: The animals used were male CD® (Sprague Dawley) IGS Rats (Charles River), injected at the age of 8 weeks and sacrificed at 11-13 weeks of age. Microglia experiment: female CX3CR1-creER/wt ; R26R-Confetti/wt (in C57/B6J background) were bred in specific-pathogen-free facility and given food and water ad libitum. Animals received tamoxifen subcutaneously at 8-9 weeks of age and underwent facial nerve transection two weeks later.
Wild animals	This study did not involve wild animals.
Field-collected samples	This study did not involve field-collected animals.