HHS Public Access

Author manuscript

Nat Mach Intell. Author manuscript; available in PMC 2022 January 01.

Published in final edited form as:

Nat Mach Intell. 2021 July ; 3(7): 590–600. doi:10.1038/s42256-021-00342-x.

Segmentation of Neurons from Fluorescence Calcium Recordings Beyond Real-time

Yijun Bao[1,*], **Somayyeh Soltanian-Zadeh**[1], **Sina Farsiu**[1,2], **Yiyang Gong**[1,3,*]

[1.] Department of Biomedical Engineering, Duke University, Durham, NC 27708

[2.] Department of Ophthalmology, Duke University Medical Center, Durham, NC 27710, USA

[3.] Department of Neurobiology, Duke University, Durham, NC 27708

Abstract

Fluorescent genetically encoded calcium indicators and two-photon microscopy help understand brain function by generating large-scale *in vivo* recordings in multiple animal models. Automatic, fast, and accurate active neuron segmentation is critical when processing these videos. In this work, we developed and characterized a novel method, Shallow U-Net Neuron Segmentation (SUNS), to quickly and accurately segment active neurons from two-photon fluorescence imaging videos. We used temporal filtering and whitening schemes to extract temporal features associated with active neurons, and used a compact shallow U-Net to extract spatial features of neurons. Our method was both more accurate and an order of magnitude faster than state-of-the-art techniques when processing multiple datasets acquired by independent experimental groups; the difference in accuracy was enlarged when processing datasets containing few manually marked ground truths. We also developed an online version, potentially enabling real-time feedback neuroscience experiments.

Introduction

Understanding of the brain has improved alongside the development of technologies that support *in vivo* optical recording of neural activity. Fluorescent genetically encoded calcium indicators[1–3] and two-photon microscopy[4–6] have enabled high-speed and large-scale recordings in multiple animal models. These experiments match neural function with neural activity by simultaneously measuring animal behavior and neural activity from hundreds to thousands of neurons. The resulting imaging movies require analysis via a multi-step process, including registration, identifying active neurons, and extracting calcium traces representing neural activities.

[*] yijun.bao@duke.edu, yiyang.gong@duke.edu.

Author contributions

Y.G. conceived and designed the project. Y.B. and Y.G. implemented the code for SUNS. Y.B. and S.S.Z. implemented the code for other algorithms for comparison. Y.B. ran the experiment. Y.B., S.S.Z., S.F., and Y.G. analyzed the data. Y.B., S.S.Z., S.F., and Y.G. wrote the paper.

Competing Interests Statement

The authors declare no competing interests.

One important but challenging step when processing fluorescence videos is to identify the individual spatial footprints of active neurons that carry the meaningful information about the animal subject's response to external or internal stimuli. While manual neuron segmentation remains the gold standard for labeling neurons, computational methods offer the benefits of consistency, scalability, and speed. Over the past decade, computational neuron segmentation methods have continually improved in speed and accuracy; algorithms now approach the accuracy of human labelers but outperform humans in speed[7, 8]. Existing automatic segmentation methods mainly fall into two classes: one class of algorithms primarily processes two-dimensional (2D) summary images as the input, while the other class primarily processes three-dimensional (3D) videos.

The first class of neuron segmentation algorithms processes small sets of 2D summary images. These images represent the temporal information of the entire video using the mean[9, 10], maximum[11], correlation[12], or a combination of these quantities[13]. Summary images can be processed by unsupervised learning methods, supervised learning methods, or a combination of both. Unsupervised learning algorithms identify pixels with neural activity and group the active pixels into neurons by identifying thresholds in intensity (e.g. ACSAT[11]) or by clustering inter-pixel correlations (e.g. HNCcorr[12]). Supervised learning algorithms (e.g. UNet2Ds[9] and Conv2D[10]), which are mainly based on convolutional neural networks (CNNs), identify similar decision boundaries after training on manually labeled examples. Some methods use unsupervised learning as a post-processing step of the supervised learning result (e.g. DISCo[13]). Overall, the class of algorithms processing 2D summary images trades off accuracy for speed. These algorithms are potentially fast for two reasons: the calculation of the summary images is fast and the processing time of the summary images is nearly independent of the video length. However, these algorithms inaccurately separate overlapping neurons because summary images blur the boundaries between neighboring neurons[7, 9, 12, 14].

The second class of neuron segmentation algorithms accurately identifies active neurons by processing blocks of frames and simultaneously analyzing both spatial features of neurons and temporal dynamics of neural activity. These methods also fall into categories of unsupervised learning, supervised learning, or a combination of both. Unsupervised learning methods model the imaging data using parameters that describe the neuron shape and temporal dynamics. These algorithms then extract neurons using optimization strategies such as principal component and independent component analysis (e.g. PCA/ICA[15]), matrix factorization (e.g. NMF[16], CNMF[17], and Suite2p[18]), or dictionary learning (e.g. SCALPEL[19]). Supervised learning methods instead seek to learn implicit features of neurons from the training data and apply the knowledge on new data. One example method (STNeuroNet[7]) in this class applied a 3D CNN to extract the spatiotemporal information. Some methods combine both learning paradigms, using unsupervised learning to propose active components and supervised learning to evaluate these components (e.g. CaImAn Batch[20]). Algorithms processing 3D inputs have produced more accurate segmentations than algorithms processing 2D summary images, but lag in speed[7]. The state-of-the-art algorithms in this class can only process recordings at a speed close to video rate, ~30 frames/s[7].

The development of neuron segmentation algorithms requires optimizing a combination of accuracy, speed, and ease of implementation targeted to specific *in vivo* neuroscience experiments. These experiments seek to perform closed-loop, real-time behavioral or neural manipulation in response to the recorded neural activity[21, 22]. Such experiments are currently out of reach for optically recorded neural activity; 2D segmentation methods (processing summary images) or 3D batch segmentation methods (processing blocks of many frames) can only find the active neurons after processing the dataset over minutes or hours, and then perform the behavioral or optical perturbations[23–26]. Online segmentation on a frame-by-frame basis is necessary for such experiments, but such detailed segmentations exert additional time pressure. Algorithms processing 2D summary images no longer achieved real-time speeds when operating frame-by-frame because they either employed substantial computation[11] or large neural networks[9, 10] for each frame. Although algorithms using 3D inputs have achieved high accuracy and video-rate processing of each data frame on average, they leave little time for image registration and computations on the neural traces needed to drive closed-loop neuroscience experiments[7]. Moreover, the creation of online versions of the 3D methods has rarely followed the development of the batch versions of the same algorithms. Existing online algorithms (e.g. ONACID[27] and CaImAn Online[20]) also lag in speed and accuracy compared to their batch counterparts.

Here, we overcome the speed-accuracy trade-off by creating Shallow U-Net Neuron Segmentation (SUNS), a new CNN-based method to segment active neurons from two-photon calcium imaging data. This method outperforms existing algorithms in four ways: (1) It is more accurate than other existing automatic methods and is on par with human labelers. (2) It is much faster than existing methods processing the 3D spatiotemporal video on a single computer. (3) It reaches high detection performance after training with few samples. (4) It generalizes high performance over various brain regions and imaging conditions. We also developed an online version that supports real-time neuron detection.

Results

Fast neuron segmentation algorithm based on a shallow U-Net.

Our premise for simultaneously improving accuracy and speed hinged on two propositions: (1) Incorporating temporal information from the video would improve accuracy; (2) Simplifying the neural network architecture would greatly improve speed. The SUNS batch algorithm (shortened to SUNS) satisfied both criteria by operating on the fluorescence recordings as a series of 2D images (Fig. 1). We incorporated temporal information into each 2D image by using temporal filtering and whitening, and we increased neuron segmentation speed by performing spatial segmentation on 2D inputs. We instantiated our temporal strategies by using traditional signal processing within the pre- and post-processing modules; these algorithms were fast and interpretable. We instantiated our spatial strategies by using a CNN within our inference module; this algorithm was fast and generalized to neuron shapes from multiple datasets.

We first maximized the signal-to-noise ratio (SNR) of active neurons in each frame by incorporating temporal information within our pre-processing module. Our previous work, STNeuroNet, incorporated temporal information by using a 3D CNN architecture

to process blocks of frames[7]. Here, we instead incorporated temporal information by applying a temporal matched filter tailored for shot noise[28] (Fig. 1b, Extended Data Fig. 1; Methods). This scheme enhanced calcium transients occurring across multiple frames, but compressed this temporal information into individual 2D frames. Additional pre-processing steps included optional spatial homomorphic filtering to remove large-scale background fluctuations, and a pixel-by-pixel whitening. Our whitening process normalized the fluorescence time series of each pixel by the estimated noise of that pixel (Methods). The resulting SNR representation of the fluorescence data highlighted active neurons and obscured inactive neurons (Supplementary Videos 1–2).

We next processed spatial information within each frame of the SNR video using a shallow U-Net (Fig. 1c, Extended Data Fig. 2). Light CNN architectures potentially offer fast training and test speeds with high generalization ability due to low overfitting. Multiple recent works developed shallow neural networks with comparable accuracy as deep networks[29–32]. We applied the same concept to separating neurons from the background (Methods). Similar to the original U-Net[33], our CNN used an encoder-decoder architecture and skip connections. Unlike a previous U-Net used for neuron segmentation[9], our network processed 2D images with a shallow structure: our CNN had a depth of 3, between 4 and 16 channels per feature map, and only one skip connection. Our network had ~5×10^3 parameters, far fewer than the 2×10^6 parameters in the original U-Net and the 9×10^5 parameters in STNeuroNet. The output of the shallow U-Net was a probability map that predicted whether each pixel came from an active neuron. Finally, our post-processing module segmented and collected neurons from the probability maps (Fig. 1d). This module thresholded probability maps, grouped active pixels from each frame into connected regions representing neurons, and merged similarly positioned neurons from all frames into unique segmentations. We determined the hyperparameters associated with these processes by maximizing the F_1 metric of the training videos (Methods).

We also developed an online version (SUNS online) to process imaging videos frame-by-frame in order to track neurons as they appeared. This capability would be critical in executing the real-time feedback experiments described above. The SUNS online procedure had two stages: It started with an initialization stage over a small number of the initial frames, and then processed the later frames one-by-one. Its implementation also differed from SUNS batch by using slightly different pre-processing and post-processing modules (Methods). The online pre-processing module calculated the signal and noise parameters over the set of initial or recent frames needed for whitening, whereas the batch pre-processing module calculated the equivalent parameters over the entire movie; the online post-processing module repeatedly updated the set of masks to include newly activated neurons, whereas the batch post-processing module aggregated all masks from the entire movie. Optional “tracking” within SUNS online provided analysis about whether detected neurons activated on a frame-by-frame basis (Methods).

SUNS segmented neurons from the Allen Brain Observatory dataset accurately and quickly.

We first evaluated the performance of our method through leave-one-out cross validation on the same set of movies and manual labels from the Allen Brain Observatory (ABO) used in our previous work (Methods)[7]. Compared with its peer neuron segmentation algorithms (STNeuroNet[7], CaImAn Batch[20], and Suite2p[18]), SUNS was both more accurate and faster (Fig. 2). Qualitatively, the segmentation results of SUNS produced shapes similar to typical neuron shapes within the ground truth (GT, *light orange*). The segmented shapes from CaImAn Batch and Suite2p were qualitatively different from the manual segmentations of the same neurons, and were sometimes non-convex or jagged (Fig. 2a–b, Supplementary Fig. 1). These irregular shapes could fall below our Intersection-over-Union (IoU) matching criterion (Methods). We quantified the neuron detection accuracy in terms of recall, precision, and F_1 for each algorithm (Methods). The F_1 of SUNS was 0.85 ± 0.02 (mean ± SD, n = 10 videos), significantly higher than other algorithms and an independent human grader (Grader 3[7]) (p = 0.002, two-sided Wilcoxon signed-rank test, n = 10 videos; Fig. 2c, Supplementary Table 1). The F_1 of SUNS was robust with moderate variations of training and post-processing parameters (Extended Data Fig. 3). Overall, our algorithm identified GT neurons better than other algorithms at nearly all SNR levels (Supplementary Fig. 2). The superior performance of SUNS remained even in the presence of intensity noise or motion artifacts (Supplementary Methods, Extended Data Fig. 4). SUNS segmented overlapping neurons at least as well as non-overlapping neurons (Extended Data Fig. 5). The temporal traces of the mean intensity from masks commonly segmented by all algorithms were qualitatively similar (Supplementary Fig. 3). Our whitening strategy, temporal filtering, and CNN architecture each contributed to this high accuracy (Extended Data Fig. 6). SUNS's human-level performance was independent of the ground truth markings (Supplementary Fig. 4).

In addition to high detection accuracy, SUNS also had high speed during both training and testing. SUNS required less training time than other methods over both a single cross-validation round and a 10-round average (Fig. 2d). The CNN component reached nearly optimal performance using 200 training images per video and converged in 200 epochs (Supplementary Fig. 5). On a single desktop, SUNS processed videos at 378 ± 2 frames/s (mean ± SD over n = 10 videos), significantly faster than all other algorithms (p = 0.002, two-sided Wilcoxon signed-rank test; Fig. 2e, Supplementary Table 1); it was also an order of magnitude faster than the 30 Hz recording frame rate of the videos. Suite2p achieved a moderate speed by reducing its computational load through temporal binning (Supplementary Fig. 6). Tests using similar shallow U-Net architectures with various resolution depths, numbers of channels in each feature map, numbers of skip connections, and loss functions revealed similar performance in speed and accuracy (Extended Data Fig. 2 and Supplementary Fig. 7).

SUNS segmented neurons from a variety of datasets accurately and quickly.

We next evaluated the generalization ability of our method using cross-validation on four other datasets: (1) training on one ABO 275 μm video and testing on nine ABO 275 μm videos; (2) training on ten ABO 275 μm videos and testing on ten ABO 175 μm videos[7];

(3) training on one Neurofinder video and testing on one paired Neurofinder video[34]; (4) training on three-quarters of one CaImAn video and testing on the remaining quarter of the same CaImAn video[20] (Methods, Supplementary Fig. 8). The F_1 scores of SUNS were nearly all significantly better than that of all other algorithms ($p = 0.23$ for STNeuroNet on ABO 175 μm, $p = 0.007$ for Suite2p on Neurofinder, and $p < 0.005$ for all other tests, two-sided Wilcoxon signed-rank test; Fig. 3a–d, Extended Data Fig. 7, Supplementary Table 2), and achieved human-level accuracy on the CaImAn dataset ($p > 0.38$, two-sided Wilcoxon signed-rank test, $n = 12$; Supplementary Fig. 9). The F_1 scores for the Neurofinder dataset had the largest variance, possibly because the training and testing videos lacked the quality or uniformity of other datasets. The biggest difference in F_1 performance between SUNS and other methods occurred when processing the CaImAn dataset, which was the dataset with the smallest number of training neurons. SUNS also performed well when trained on a group of datasets with different imaging conditions (Supplementary Fig. 10). While transfer learning could potentially improve segmentation performance[35], it did not enhance performance of SUNS (Supplementary Fig. 11).

For the four datasets, SUNS was significantly faster than all other algorithms ($p < 0.002$, two-sided Wilcoxon signed-rank test, Fig. 3a–d, Supplementary Table 2). The frame rates when testing the CaImAn movies (Fig. 3d) were much higher than the rates when testing the other videos because the lateral dimensions of each CaImAn video we used were approximately one-half of the dimensions of the other videos. The segmentation results of SUNS were closer to the ground truth (*light orange*) in both numbers and shapes than the segmentation results of other methods (Fig. 3e–f, Supplementary Fig. 1).

SUNS online also segmented neurons from the ABO dataset accurately and quickly.

We evaluated the performance of SUNS online using the same ABO 275 μm dataset, and compared our method with its peer algorithm, CaImAn Online[20]. SUNS online was both more accurate and faster than CaImAn Online (Fig. 4). Qualitatively, the segmentation results of SUNS online produced shapes similar to GT neuron shapes (*light orange*); neurons identified by CaImAn Online were sometimes non-convex or jagged, and could fall below our IoU matching criterion (Fig. 4a–b). Quantitatively, the F_1 of SUNS online was 0.85 ± 0.01 (mean ± SD over $n = 10$ videos), significantly better than CaImAn Online and Grader 3 on the ABO 275 μm dataset ($p = 0.002$, two-sided Wilcoxon signed-rank test, $n = 10$ videos; Fig. 4c, Supplementary Table 3). The F_1 scores produced by SUNS online when operating on the other datasets used in Fig. 3 were also significantly higher than the F_1 scores produced by CaImAn Online ($p < 0.002$, two-sided Wilcoxon signed-rank test; Extended Data Fig. 8, Supplementary Table 4). Lastly, the accuracy of SUNS online using the trained CNN models and hyper-parameters from the batch training was higher than the accuracy of SUNS online using the CNN models and hyper-parameters independently trained within the online pre- and post-processing pipeline (Methods, Supplementary Fig. 12). Thus, SUNS online can employ parameters from the batch version without retraining, leading to straightforward implementation when attempting real-time experiments on novel datasets.

In addition to higher accuracy, SUNS online was also significantly faster than CaImAn Online on the ABO 275 μm dataset ($p = 0.002$, two-sided Wilcoxon signed-rank test, $n = 10$ videos; Fig. 4d, Supplementary Table 3) and on the other datasets used in Fig. 3 ($p < 0.002$, two-sided Wilcoxon signed-rank test; Extended Data Fig. 8, Supplementary Table 4). SUNS online processed the ABO 275 μm videos at 117 ± 7 frames/s (mean ± SD, $n = 10$ videos) on a single desktop computer, and thus was on average multiple times faster than the recording frame rate of the video. SUNS online processed every frame faster than the imaging rate, consistently processing most frames in ~9 ms (Fig. 4e, Supplementary Fig. 13). Changing the frequency of updating the neuron masks modulated the trade-offs between the algorithm's response time to new neurons and the algorithm's accuracy and speed (Extended Data Fig. 9). Applying optional baseline and noise updates further increased the accuracy at the cost of slightly lower average speed, (Methods, Extended Data Fig. 10). SUNS online with the "tracking" option indicated whether neurons were active in every frame after they were initially detected; such outputs would provide frame-by-frame indicators of activity from detected neurons (Supplementary Videos 1–2). However, enabling this option lowered both speed and accuracy (Supplementary Fig. 14).

Discussion

We present in this paper an automated, fast, and accurate active neuron segmentation method using three modules. We used pre-processing to incorporate temporal information into each 2D image and suppress noise; we used a shallow U-Net to quickly identify features of active neurons; we used post-processing to finalize the spatial footprints of consistently detected neurons. SUNS not only had high accuracy, but also was much faster than the video acquisition rate; such "beyond real-time" speeds can support a critical set of neuroscience experiments requiring intricate calculations and feedback between individual imaging frames. The batch version of SUNS can enable experimentalists to quickly obtain the activity profiles of individual neurons soon after the dataset acquisition and construct additional perturbative trials with the same animal. SUNS batch can process movies at 3 ms per frame on a single desktop computer, bypassing the need for additional parallelization on computer clusters. SUNS online can process recordings on a frame-by-frame basis within a fraction of the frame acquisition time on a single desktop as well; it opens additional time within each frame acquisition to support online electrical, optogenetic, or behavioral feedback.

Compared to our previous work, STNeuroNet, the scheme in SUNS incorporating temporal information through traditional signal processing was simple but effective. The success of such simple schemes suggests that substantial gains in speed and accuracy can arise from highlighting the natural temporal features within specific datasets, such as the calcium transient within two-photon imaging movies. This temporal filtering scheme and SUNS's natural ability to temporally screen multiple instances of the same neurons over multiple frames both partly contributed to SUNS's high accuracy.

Compared to the original U-Net, the CNN of SUNS is light-weight. This CNN still effectively learned the features of active neurons from the SNR images likely because the task of segmenting neurons from the background in a single frame is relatively easy

compared to other biomedical image processing tasks[36]. The small numbers of channels per feature map effectively captured the spatial features of active neurons. Such features were effectively integrated by the shallow depth network; each pixel in the lowest-resolution layer integrated information from 23 × 23 pixels of the input image, an area large enough to cover a typical neuron of radius 6–10 pixels in our datasets.

SUNS's simple network architecture enabled not only high processing speeds but also high generalization ability[32]. Such generalization ability appeared as SUNS's consistently first-in-class accuracy when processing multiple datasets acquired by independent labs. Generalization ability also appeared as SUNS's high accuracy relative to peer algorithms when training on a limited dataset, such as three-quarters of a video within the CaImAn dataset. SUNS achieved such generalization ability likely because the small number of network parameters prevented overfitting. The spatial discrimination module thus restricted learning to the most fundamental features of active neurons; it effectively ignored features specific to the training neurons but not generalizable to the test neurons.

End-users can quickly customize SUNS to process imaging data from new brain regions or specialized microscopes for two reasons: (1) SUNS produces high accuracy when trained on a small subset of frames from one video. The production of manual annotations at this scale and training SUNS on a commercial desktop should take less than 4 hours. (2) Users can directly apply the CNN models and optimal hyperparameters from the batch version to the online version. Other matched batch and online algorithms[20] do not share parameters and CNN models, and thus require retraining when implementing the online version.

The online version of SUNS underperformed the batch version in accuracy and speed. SUNS online was less accurate because it does not access the full dataset to optimally segment and combine multiple occurrences of the same neuron. SUNS online was slower because it could not efficiently parallelize execution and memory transfers between the CPU and GPU. Future improvements in the hardware data transfer mechanics between the computer memory and graphics memory and in the software execution mechanics of matrix-based calculations will potentially help SUNS online perform at a speed close to the speed of SUNS batch.

Our algorithm informs the future development of similar algorithms for one-photon fluorescence imaging videos. One-photon images typically have worse spatial resolution than two-photon images due to one-photon imaging's poor optical sectioning and its susceptibility to scattering within tissue. These poorer imaging metrics result in high out-of-focus fluorescence background fluctuations and large spatial overlap between neurons. Accounting for such differences paved the way for the non-negative matrix factorization family of algorithms that process mostly high spatial resolution two-photon videos to also process low spatial resolution one-photon calcium recordings in scattering tissue[37]. SUNS can similarly overcome such challenges in three ways. First, SUNS can segment overlapping neurons from the frames when they are activated alone. Additional watershed segmentation[38] could increase the accuracy when separating overlapping neurons. Second, SUNS already implements a form of spatial filtering for background removal. Additional feature channels will likely account for the spatially-varying out-of-plane fluorescence[37].

Third, adjustments to the consume ratio and IoU metrics will likely better distinguish neurons imaged at lower resolution.

SUNS is a fast and accurate neuron segmentation algorithm, but neuron segmentation is only a part of calcium imaging analysis pipeline. Matching peer pipelines[18, 20], a complete pipeline incorporating SUNS should also employ an image registration module to correct motion artifacts[39] and a trace unmixing module to decontaminate crosstalk from neighboring calcium sources[40]; SUNS can work in conjunction with existing implementations of both types of algorithms. For example, if we incorporated the registration algorithm of Mitani et al. that reported a speed of 2 ms/frame on older hardware[41], our overall pipeline would still be much faster than the video rate.

Beyond specific applications in fluorescence calcium imaging, our work is broadly relevant to the machine learning community because it adds a data point to the on-going discussion about neural network dimensionality. Current hypotheses about "lottery-ticket" networks[42] and replication of neural network function[30, 43] corroborate the results of this work; they suggest that simple network structures with few weights can recapitulate the accuracy of complex network structures with many weights. However, attaining these sparse networks requires the training of an intermediate dense network that either serves as a model for additional pruning or a model for the sparse network to mimic. Here, we demonstrate that a shallow and light CNN, with appropriate pre-processing, is sufficient for neuron segmentation tasks. Such success suggests that starting computer vision projects with the most parsimonious algorithms could save implementation, design, training, and test time for a class of simple computer vision problems.

Materials and Methods

Active neuron segmentation.

SUNS contained three major components: an optional pre-processing module made two-photon microscopy data appropriate for CNN analysis, a CNN inference module predicted the probability of whether each pixel came from an active neuron in the frame, and a post-processing module segmented and collected neurons from the probability maps.

Pre-processing steps.—All videos used in our work were previously registered. Our pre-processing applied optional spatial filtering, temporal filtering, and whitening to reduce noise and enhance active neurons. First, we optionally corrected for large-scale background fluctuation and non-uniform background illumination by applying spatial homomorphic filtering[44] to each frame of the video. Given a raw intensity image $I(x,y)$, the spatially filtered image $I'(x,y)$ was

$$I'(x,y) = \mathrm{SF}[I(x,y)] = \exp\{\log[I(x,y)+1] - \log[I(x,y)+1]*G(x,y)\},$$

where $G(x,y)$ was a 2D Gaussian kernel with standard deviation of 40 μm, and "*" denotes spatial convolution. Second, we enhanced the SNR of calcium transient waveforms by applying a temporal matched filter that accounts for shot-noise[28]. Given the spatially filtered fluorescence time series $I'(x,y,t)$, the temporally filtered time series was

$$I''(x,y,t) = \mathrm{TF}[I'(x,y,t)] = I'(x,y,t) \circ h(t),$$

where "∘" denotes pixel-by-pixel temporal convolution, and the filter kernel $h(t)$ was the time-reversed average fluorescence response of the GT neurons to calcium transients with moderate peak SNR between 5 and 8 aligned to their peaks; these transients likely represent the temporal response to single action potentials[2]. We also tested the algorithm performance when using a simple exponentially decaying kernel with the decay time reported in the literature[2]. Third, we highlighted active neurons and de-emphasized inactive neurons by converting the result to an SNR video. This process normalized the fluorescence time series of each pixel by the estimated noise of that pixel. Given the temporally filtered fluorescence time series $I''(t)$ of each pixel, the SNR time series was

$$\mathrm{SNR}[I''(t)] = \frac{I''(t) - \mathrm{M}[I''(t)]}{\sigma[I''(t)]},$$

where $\mathrm{M}[I''(t)]$ was its median intensity representing the baseline, and $\sigma[I''(t)]$ was the quantile-based noise, calculated as

$$\sigma[I''(t)] = \{\mathrm{M}[I''(t)] - \mathrm{Q}_1[I''(t)]\} / \left[\sqrt{2}\,\mathrm{erf}^{-1}(0.5)\right],$$

where $\mathrm{Q}_1[I''(t)]$ is the first quantile and erf^{-1} is the inverse error function. The noise calculation is similar to median-based standard deviation[45]. We used the median baseline and the quantile-based noise because these metrics deemphasize fluctuations induced by calcium activity, and thus appropriately represented the fluorescence variation of the neuron at rest.

Neural network architecture.—We processed each pre-processed SNR frame using a 2D shallow U-Net. Similar to the original U-Net[33], our network structure also used an encoder-decoder architecture and skip connections. In our final version, we used two convolution/max-pooling blocks to reduce dimensionality and two convolution/transpose-convolution (deconvolution) blocks to restore input dimensionality. The number of channels in each feature map was 4, 8, and 16, respectively at each resolution depth. We used a skip connection to concatenate the output of the first convolution layer and the output of the decoder; this connection enabled the network to extract both low-level and high-level information. An Exponential Linear Unit (ELU) activation and dropout rate of 0.1 (0.2 at the deepest depth) followed each convolutional layer, except in the last output layer; that layer transformed the feature map to an output 2D probability map by using a sigmoid activation function without dropout. By minimizing the network depth, the number of channels per feature map, and the number of skip connections, our network only trained ~5×10^3 parameters.

Post-processing steps.—After CNN inference, we segmented the final binary neuron masks by applying post-processing to the output probability maps. The post-processing steps included probability thresholding, spatial grouping within each frame, and temporal merging

across all frames. First, we determined the active pixels in each frame by binarizing the probability maps with a probability threshold (th_{prob}). Second, we grouped neighboring active pixels within a frame into connected active regions. We also removed false positive active regions with areas smaller than a minimum area (th_{area}). Finally, we aggregated all identified active masks from different frames to obtain the segmentation results for the entire video. We merged similarly positioned active masks from different frames because they likely belonged to the same neuron. The similarly positioned masks satisfied one of three requirements: (1) close center of masses (COMs), (2) large IoU, or (3) large consume ratio. The threshold of the COM distance (th_{COM}) was smaller than the typical neuron radius. For two binary masks, m_1 and m_2, we defined the IoU and consume ratio respectively as

$$\text{IoU}(m_1, m_2) = \frac{|m_1 \cap m_2|}{|m_1 \cup m_2|},$$

and

$$\text{consume}(m_1 | m_2) = \frac{|m_1 \cap m_2|}{|m_2|}.$$

We merged pairs of masks that had $\text{IoU}(m_1, m_2) \geq 0.5$, $\text{consume}(m_1|m_2) \geq 0.75$, or $\text{consume}(m_2|m_1) \geq 0.75$. When the mask pairs had a COM distance below th_{COM} or IoU satisfying the above condition, we combined the masks with addition. When the mask pairs satisfied one of the above consume ratio conditions, we eliminated the larger mask of the pair if the area of the larger mask was larger than the maximum neuron size; otherwise, we added the masks. We merged the neurons with close COMs first, then those with large IoU, and finally those with large consume ratio. After aggregating all active masks, we further selected the regions of interest (ROIs) that lasted for a minimum number of consecutive frames (th_{frame}). Finally, we binarized each merged ROI using half of the maximum value of that ROI as the threshold, which produced the final masks of the segmented active neurons.

Online processing.—As opposed to SUNS batch that processed the entire movie all together, SUNS online contains two stages: It initialized on a limited number of the initial frames (the first n_{init} frames) of the movie, and then processed the following data frame-by-frame. SUNS batch and SUNS online also used different pre-processing and post-processing modules. The online pre-processing module initialized or updated the distribution of the fluorescence with a limited number of recent frames in order to whiten frames as they are acquired, rather than computing the distribution once at the end of the movie. In the initialization stage, we used the SUNS batch pipeline to analyze the initialization frames and compute the following set of outputs: (1) the median baseline and quantile-based noise of all pixels; (2) both the binary masks and the real-number masks of all neuron candidates even if they were active for less than th_{frame} frames; (3) the indices of active frames for each neuron candidate. During the frame-by-frame processing stage, the pre-processing module used the same spatial and temporal filtering as the batch version, but performed SNR normalization using the median baseline and quantile-based noise determined from the initialization frames rather than those determined from the entire video calculated at

the end of the movie. Because the baseline and noise may vary after the initial frames, we developed an optional baseline and noise update that corrects the baseline and noise estimates through the duration of imaging movies. Every ~n_{init} frames, we saved a buffer of the recent n_{init} images after temporal filtering but before SNR normalization, and calculated the median baseline and quantile-based noise of these images. Because this calculation was computationally intensive, we distributed the calculation over multiple frames throughout the online processing: after pre-processing every frame, we only calculated the median baseline and quantile-based noise of a column or a part of a column. After calculating the median baseline and quantile-based noise of all pixels, we then applied the new baseline and noise images in the following ~n_{init} frames.

The online post-processing module needed to aggregate the newly active masks from the recently processed frames in an online way rather than aggregating all masks at the end of the entire movie. We instantiated this process after initialization by examining every block of n_{merge} frames after the process, locating the newly detected neuron candidates in these frames, and finally merging these new candidates with previously detected neuron candidates. When merging the new neurons with existing neurons, we applied the same conditions on the IoU and consume ratio as in the batch version, but we ignored the COM distance requirement and used simple addition to merge neuron candidates. We then added newly found neurons that did not match existing neurons into the set of detected neurons. We kept all candidate neurons throughout the online processing regardless of whether they satisfied the consecutive frame requirement, but we also screened the candidates with th_{frame} after every merge to create the set of active neurons as the intermediate output. After the online processing, we performed an extra merging step for all the identified neurons according to the IoU and consume ratio, in case a neuron was split into two highly overlapped masks based on their early appearances.

We also developed a “tracking” option that used a different neuron aggregation strategy in the post-processing module within the frame-by-frame stage that tracked the activation of existing neurons at each frame. This option provided a frame-by-frame readout of the neural activity from detected neurons, information that could determine future closed-loop neural or behavioral feedback. We checked whether active regions within each frame matched to existing neurons using the same IoU and consume ratio metrics. If they matched existing neurons, we marked the matched existing neurons as active for that frame. At the end of each consecutive active period of each existing neuron, we collected all the newly segmented candidate masks associated with that neuron, added them to the existing real-number mask of that neuron, and updated the binary mask of that neuron. If the active masks did not match existing neurons, we merged these newly segmented candidate masks after every n_{merge} frames and added the result to the set of existing neurons as in the standard online version.

Training and parameter optimization.—We found the optimal CNN model and post-processing hyperparameters by training. To create GT labels for each frame, we first applied FISSA[40] to decontaminate the fluorescence traces of each GT neuron mask. We then calculated the SNR traces of each neuron by normalizing each trace: we subtracted each trace by the trace median and then divided the resulting trace by the quantile-based noise.

When the SNR trace was higher than a threshold (th_{SNR}) at a given time, we treated this neuron as active at that time. We then added the masks of all active neurons at each frame to create the binary GT labels for each frame of the movie. We performed these steps only once for each video and used these results during subsequent rounds of cross-validation. Thus, the average training time for each round within a 10-round cross-validation was shorter than that for a single round of cross-validation.

We selected $1800/n$ images at uniform intervals from each training video, where n is the number of training videos, to produce 1800 images for training in each round of cross-validation. We fed the GT labels and their contemporaneous SNR images into the CNN for training. We trained the CNN for 200 epochs, with random flips and rotations for each pair of SNR image and GT mask. We used the Adam optimizer with a 0.001 learning rate and 20 batch size. We used one of two types of loss functions: a weighted sum of Dice loss[46] and binary cross-entropy, or a weighted sum of Dice loss and focal loss[47]. To determine the optimal post-processing hyperparameters, we first computed the segmentation results of all training videos using various th_{prob}, th_{area}, th_{COM}, and th_{frame}, and then determined the optimal thresholds that yielded the highest mean F_1 score on the training set. We applied the same trained CNN models and post-processing hyperparameters to both the batch version and the online version. We also tested SUNS online using the trained CNN models and hyperparameters that resulted from applying the pre- and post-processing procedures from SUNS online.

Datasets and characterization scheme.

Allen Brain Observatory dataset ——The ABO dataset we used included 10 videos recorded from 275 μm deep in the primary visual cortex (VISp) of 10 mice and another 10 videos recorded from 175 μm deep in the VISp of 10 different mice using two-photon microscopy; all mice expressed the GCaMP6f calcium sensor[48]. Each video lasted about 13 min at a frame rate of 30 frames/s. Each frame was cropped to 487 × 487 pixels. We used our previous manual annotations[7] as the GT neuron markings containing ~200 – 400 GT neurons in each video. This GT set was created by combining the labels of two experienced graders (Graders 1&2), and both graders had access to the markings originally provided by ABO. Grader 3, who did not have access to any directly related markings, served as an inter-human reliability test and a performance baseline.

We used this dataset to evaluate the performance of all neuron segmentation algorithms in three types of cross-validation: (1) ten rounds of leave-one-out cross-validation (training on nine 275 μm videos and testing on the remaining one 275 μm video); (2) ten rounds of 1-to-9 generalization test (training on one 275 μm video and testing on the remaining nine 275 μm videos); (3) one round of generalization test on different recording depths (training on all ten 275 μm videos and testing on all ten 175 μm videos). In each round, we used the training video(s) to train the CNN, searched for the optimal post-processing hyperparameters, and applied the trained CNN along with their optimal hyperparameters to the remaining test video(s). We used each video of the 275 μm set as the test video exactly once in test (1) and as the training video exactly once in test (2). In test (2), we averaged the nine results from the nine test videos to calculate the metrics for each round of cross-validation. We

optimized the SUNS post-processing parameters over the range of values in Supplementary Table 6. When running SUNS online, we set n_{init} to be 900 frames (30 s) and n_{merge} to be 30 frames (1 s). We optimized the parameters of all other algorithms over the range of values in Supplementary Tables 7–10.

Neurofinder dataset ——We used 12 two-photon calcium imaging videos (sets 01, 02, and 04) from the Neurofinder dataset[34]. We used the GT neuron markings associated with these videos made in our previous work[7]. Each set of videos imaged neurons expressing GCaMP6s. The three sets of videos originated from three independent labs, and each set generally imaged from one cortical or subcortical brain region. Each set of videos contained two "train" videos and two "test" videos. While each set of videos had different imaging conditions (e.g. different numbers of frames, numbers of pixels in each frame, or recording frame rate), one "train" video and one "test" video within each set had matching imaging conditions. This structure resulted in 6 pairs of train-test videos matched to the same serial number (e.g. 04.00 and 04.00.test). We used each of the 12 videos as the training video exactly once within a 12-round, one-to-one cross-validation. In each round, we used the one video within a train-test pair to train the CNN and search for the optimal post-processing hyperparameters. We then tested the trained CNN and the associated optimal hyperparameters on the remaining video within the same train-test pair. We optimized the SUNS post-processing parameters over the range of values in Supplementary Table 6. When running SUNS online, we set n_{init} to be 300 frames (37.5–100 s) and n_{merge} to be 10 frames (1.25–3.33 s). We optimized the parameters of all other algorithms over the range of values in Supplementary Tables 7–10. We adjusted the magnification of the 01 series to 0.8 μm/pixel and that of the 02 series to 1.15 μm/pixel. Without this adjustment, the ground truth neurons would report the area of layer 2/3 pyramidal neurons in the primary visual cortex of mice as 2–2.5× their area reported in the literature for the 01 series and 0.6–0.7× for the 02 series[49, 50].

CaImAn dataset ——We used four two-photon calcium imaging videos from the CaImAn dataset (J115, J123, K53, and YST) that did not come from Neurofinder, as well as the associated GT neuron markings from that publication[20]. These four videos originated from two independent labs, and they were recorded from three different brain regions with three different calcium sensors and imaging conditions. We divided each video into a set of 4 quarter-sized sub-videos with equal size and similar numbers of neurons. We ensured spatial separation between the sub-videos by removing pixels near the sub-video boundaries; the distance between each quarter was at least 15 pixels. We performed 4 rounds of leave-one-out cross-validation within each set of 4 sub-videos originating from the same full-sized video: we used 3 out of the 4 sub-videos to train the CNN, searched for the optimal post-processing hyperparameters, and applied the trained CNN along with their optimal hyperparameters to the remaining test sub-video. To compensate for the smaller image size, we selected 4× the number of images from each video (7200 total images) for training the CNN compared to other datasets. We optimized the SUNS post-processing parameters over the range of values in Supplementary Table 6. Because the magnifications of imaging systems were not given in these datasets, we assumed that the magnifications were proportional to the average neuron radius provided in the header information of the

dataset. When running SUNS online, we set n_{init} to correspond with 30 s (300 frames for YST and 900 frames for the other videos) and n_{merge} to correspond with 1 s (10 frames for YST and 30 frames for the other videos). We optimized the parameters of all other algorithms over the range of values in Supplementary Tables 7–10.

Evaluation metrics.

We evaluated all segmentation methods by comparing their results with the manual GT labels. We assessed each algorithm by quantifying three neuron detection metrics: recall, precision, and F_1 score, defined as follows:

$$\text{Recall} = \frac{N_{\text{TP}}}{N_{\text{GT}}},$$

$$\text{Precision} = \frac{N_{\text{TP}}}{N_{\text{detected}}},$$

$$F_1 = \frac{2}{\text{Recall}^{-1} + \text{Precision}^{-1}},$$

where N_{TP}, N_{GT}, and N_{detected} are the numbers of true positive neurons, GT neurons, and detected neurons by the method, respectively. We used the IoU metric along with the Hungarian algorithm to match the masks between the GT labels and the detected masks[20]. We calculated the distance Dist between any pair of masks from the GT set (m_i^{GT}) and the detected set (M_j) as

$$\text{Dist}\left(m_i^{GT},\ M_j\right) = \begin{cases} 1 - \text{IoU}\left(m_i^{GT},\ M_j\right), & \text{IoU}\left(m_i^{GT},\ M_j\right) \geq 0.5 \\ 2, & \text{IoU}\left(m_i^{GT},\ M_j\right) < 0.5\,. \end{cases}$$

In the above equation, Dist = 2 denotes masks that cannot be matched due to their small IoU score. Next, we applied the Hungarian algorithm to solve the linear assignment problem using the above distance matrix and denoted the matched masks whose distances were smaller than 2 as true positive masks.

Hardware and speed analysis.

All algorithms were evaluated on a single computer (Windows 10, AMD 1920X CPU, 128 GB RAM, NVIDIA Titan RTX GPU). The processing time measurement started after all the data was loaded into memory, so the hard drive reading and writing time was excluded (except for a portion of STNeuroNet execution, Supplementary Methods). The training times of all methods were estimated from the creation time of training output files.

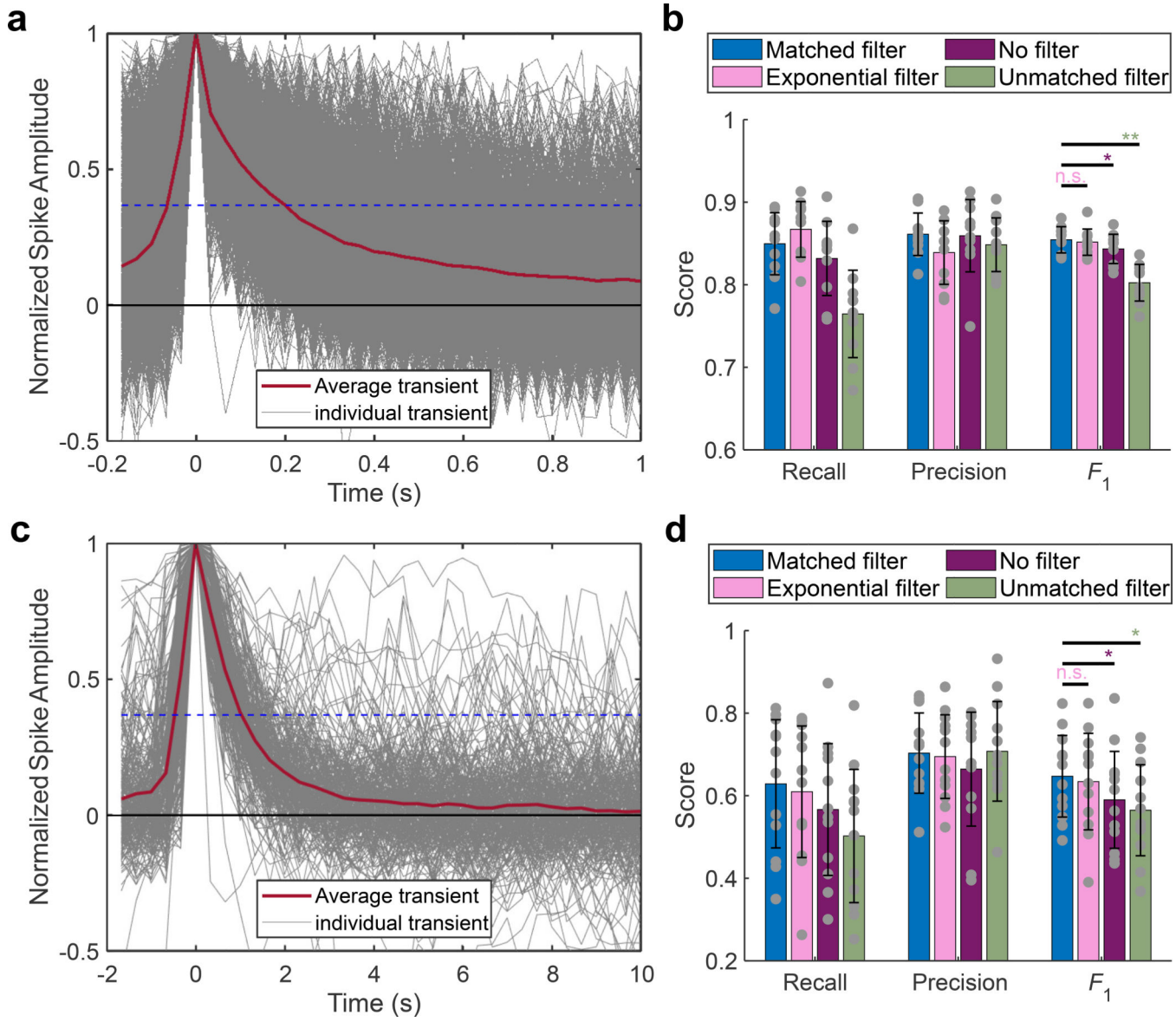
Code availability.

Code for SUNS can be accessed in https://github.com/YijunBao/Shallow-UNet-Neuron-Segmentation_SUNS [51]. The version to reproduce the results in this paper can be accessed in https://github.com/YijunBao/SUNS_paper_reproduction [52].

Data availability.

The trained network weights, and the optimal hyperparameters can be accessed in https://github.com/YijunBao/SUNS_paper_reproduction/tree/main/paper_reproduction/training%20results. The output masks of all neuron segmentation algorithms can be accessed in https://github.com/YijunBao/SUNS_paper_reproduction/tree/main/paper_reproduction/output%20masks%20all%20methods. We used three public datasets to evaluate the performance of SUNS and other neuron segmentation algorithms. We used the videos of ABO dataset from https://github.com/AllenInstitute/AllenSDK/wiki/Use-the-Allen-Brain-Observatory-%E2%80%93-Visual-Coding-on-AWS, and we used the corresponding manual labels created from our previous work, https://github.com/soltanianzadeh/STNeuroNet/tree/master/Markings/ABO. We used the Neurofinder dataset from https://github.com/codeneuro/neurofinder, and we used the corresponding manual labels created from our previous work, https://github.com/soltanianzadeh/STNeuroNet/tree/master/Markings/Neurofinder. We used the videos and manual labels of CaImAn dataset from https://zenodo.org/record/1659149. A more detailed description of how we used these dataset can be found in the readme of https://github.com/YijunBao/SUNS_paper_reproduction/tree/main/paper_reproduction.

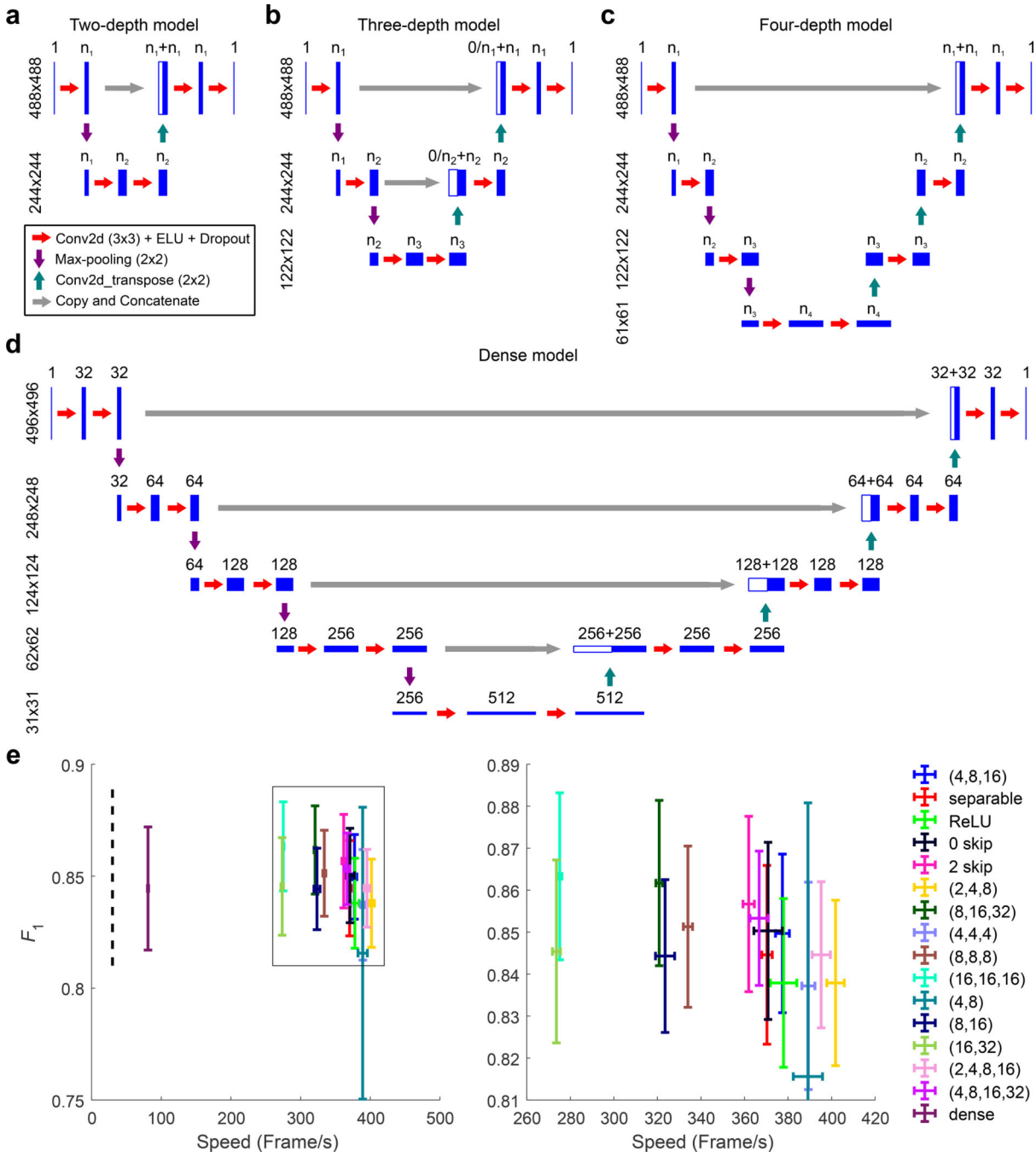
Extended Data



Extended Data Fig. 1. The average calcium response formed the temporal filter kernel.

We determined the temporal matched filter kernel by averaging calcium transients within a moderate SNR range; these transients likely represent the temporal response to single action potentials[2]. (a) Example data show all background-subtracted fluorescence calcium transients of all GT neurons in all videos in the ABO 275 μm dataset that showed peak SNR (pSNR) in the regime 6 < pSNR < 8 (*gray*). We minimized crosstalk from neighboring neurons by excluding transients during time periods when neighboring neurons also had transients. We normalized all transients such that their peak values were unity, and then averaged these normalized transients into an averaged spike trace (*red*). We used the portion of the average spike trace above e^{-1} (*blue dashed line*) as the final template kernel. (b) When analyzing performance on the ABO 275 μm dataset through 10-fold leave-one-out

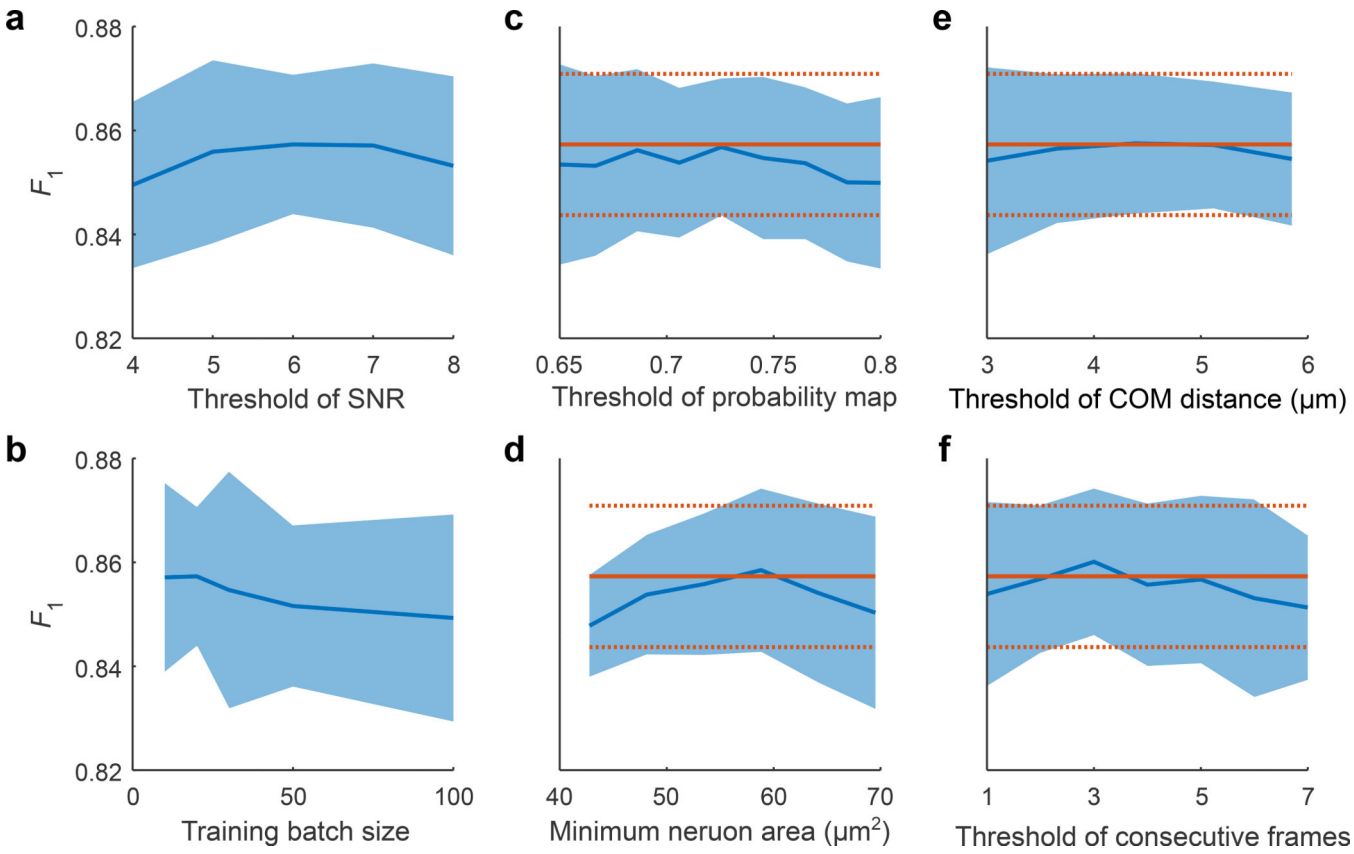
cross-validation, using the temporal kernel determined in (a) within our temporal filter scheme achieved significantly higher F_1 score than not using a temporal filter or using an unmatched filter (*$p < 0.05$, **$p < 0.005$; two-sided Wilcoxon signed-rank test, $n = 10$ videos) and achieved a slightly higher F_1 score than using a single exponentially decaying kernel ($p = 0.77$; two-sided Wilcoxon signed-rank test, $n = 10$ videos). Error bars are standard deviations. The gray dots represent scores for the test data for each round of cross-validation. The unmatched filter was a moving-average filter over 60 frames. (c-d) are analogous to (a-b), but for the Neurofinder dataset. We determined the filter kernel using videos 04.01 and 04.01.test.



Extended Data Fig. 2. The complexity of the CNN architecture controlled the tradeoff between speed and accuracy.

We explored multiple potential CNN architectures to optimize performance. (a-d) Various CNN architectures having depths of (a) two, (b) three, (c) four, or (d) five. For the three-depth architecture, we also tested different numbers of skip connections, ReLU (Rectified Linear Unit) instead of ELU (Exponential Linear Unit) as the activation function, and separable Conv2D instead of Conv2D in the encoding path. The dense five-depth model mimicked the model used in UNet2Ds[9]. The legend "0/n_i+n_j" represents whether the skip connection was used (n_i+n_j) or not used (0+n_j). (e) The F_1 score and processing speed

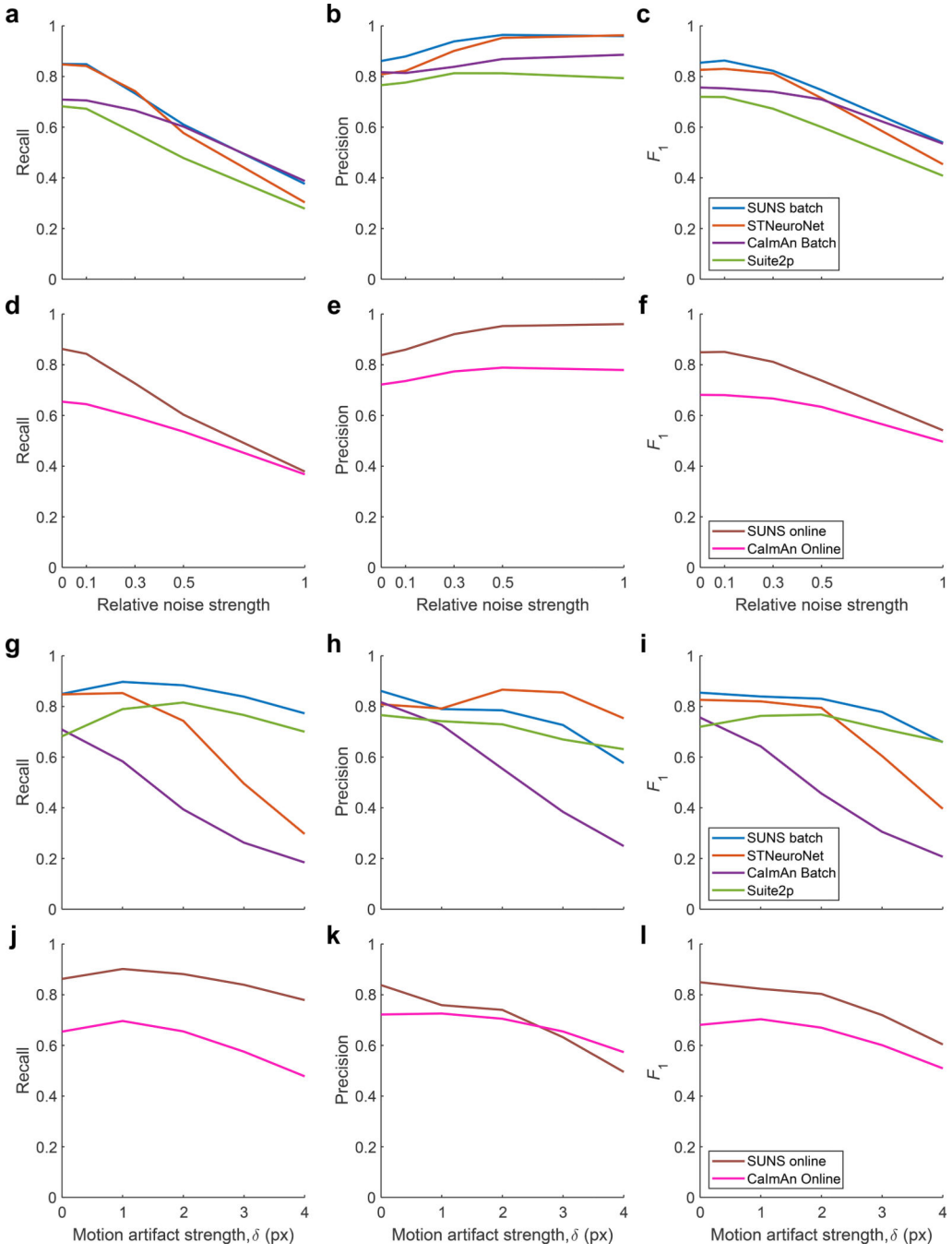
of SUNS using various CNN architectures when analyzing the ABO 275 μm dataset through 10-fold leave-one-out cross-validation. The right panel zooms in on the rectangular region in the left panel. Error bars are standard deviations. The legend (n_1, n_2, …, n_k) describes architectures with k-depth and n_i channels at the i^{th} depth. We determined that the three-depth model, (4,8,16), using one skip connection at the shallowest layer, ELU, and full Conv2D (Fig. 1c), had a good trade-off between speed and accuracy; we used this architecture as the SUNS architecture throughout the paper. One important drawback of the ReLU activation function was its occasional (20% of the time) failure during training, compared to negligible failure levels for the ELU activation function.



Extended Data Fig. 3. The F_1 score of SUNS was robust to moderate variation of training and post-processing parameters.

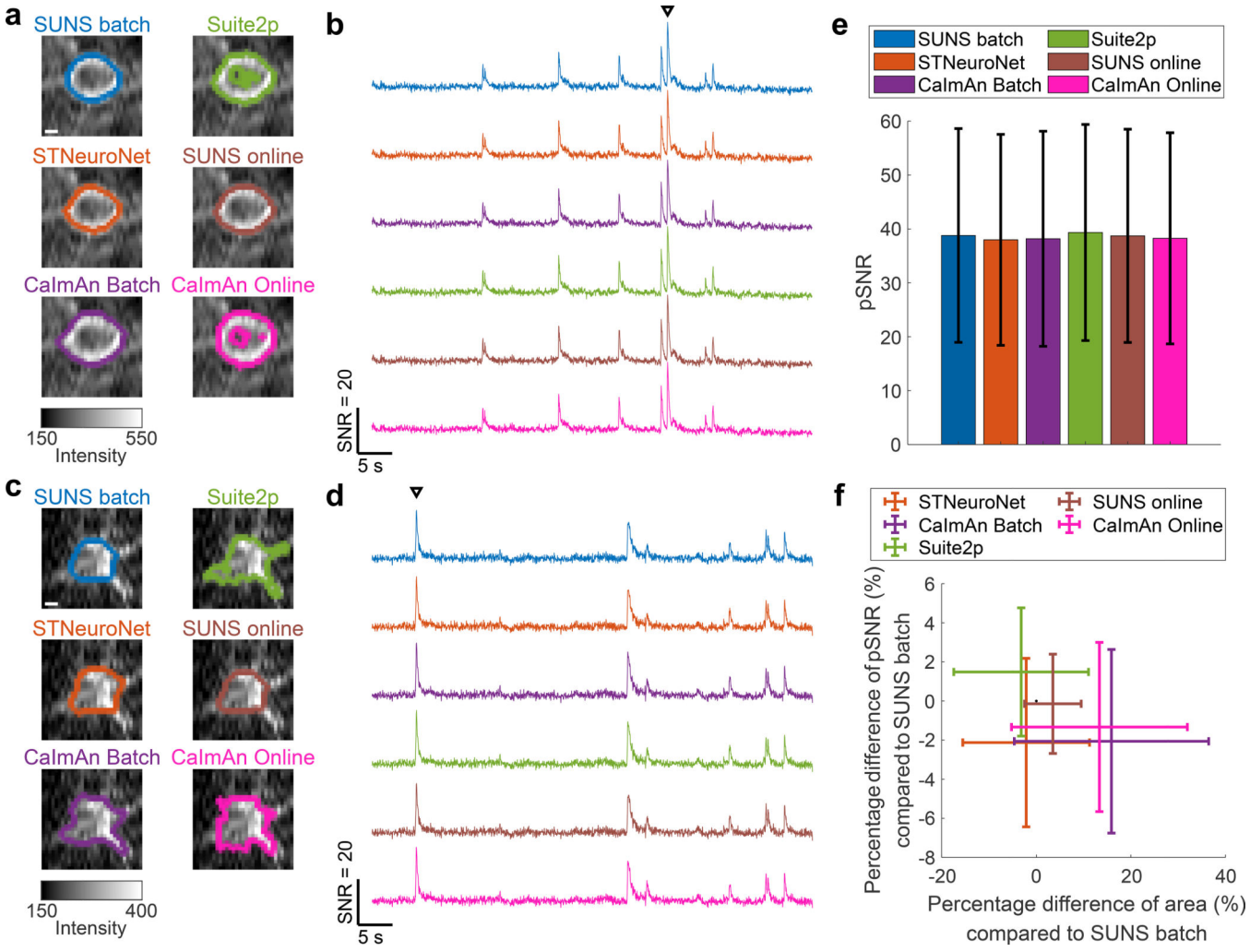
We tested if the accuracy of SUNS when analyzing the ABO 275 μm dataset within the 10-fold leave-one-out cross-validation relied on intricate tuning of the algorithm's hyperparameters. The evaluated training parameters included (a) the threshold of the SNR video (th_{SNR}) and (b) the training batch size. The evaluated post-processing parameters included (c) the threshold of probability map (th_{prob}), (d) the minimum neuron area (th_{area}), (e) the threshold of COM distance (th_{COM}), and (f) the minimum number of consecutive frames (th_{frame}). The solid blue lines are the average F_1 scores, and the shaded regions are mean ± one standard deviation. When evaluating the post-processing parameters in (c-f), we fixed each parameter under investigation at the given values and simultaneously optimized the F_1 score over the other parameters. Variations in these hyperparameters produced only

small variations in the F_1 performance. The orange lines show the F_1 score (*solid*) ± one standard deviation (*dashed*) when we optimized all four post-processing parameters simultaneously. The similarity between the F_1 scores on the blue lines and the scores on the orange lines suggest that optimizing for three or four parameters simultaneously achieved similar optimized performance. Moreover, the relatively consistent F_1 scores on the blue lines suggest that our algorithm did not rely on intricate hyperparameter tuning.



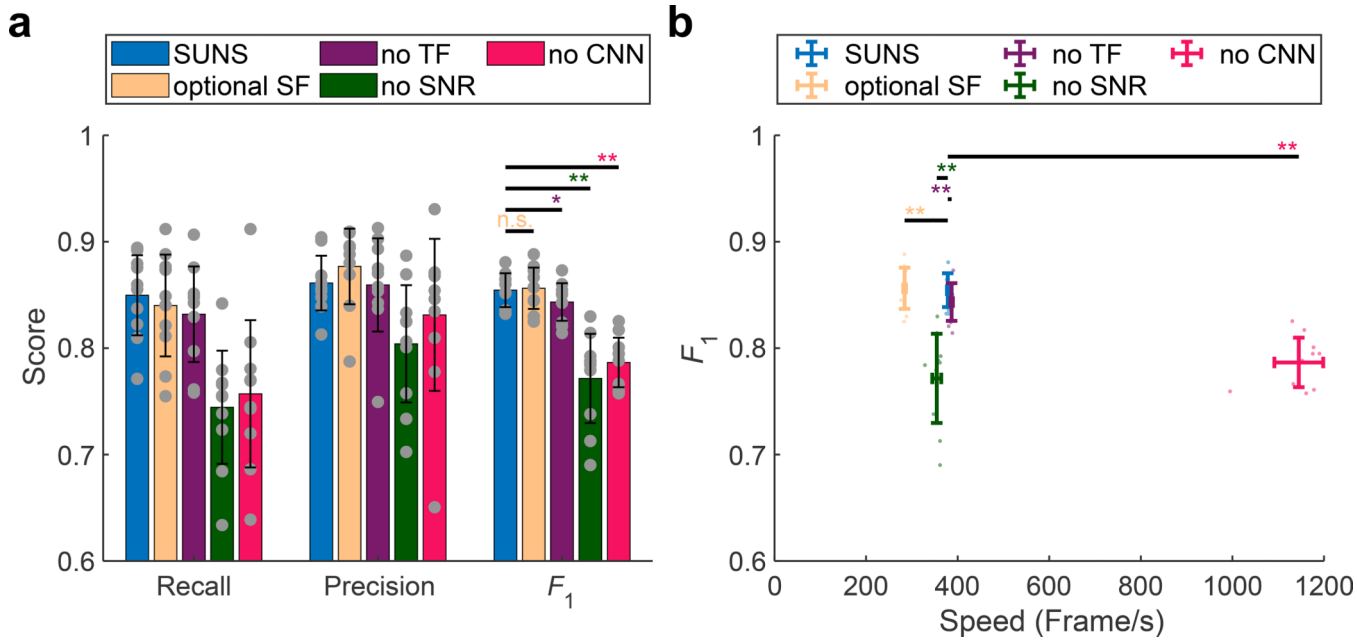
Extended Data Fig. 4. The performance of SUNS was better than that of other methods in the presence of intensity noise or motion artifacts.

The (a, d) recall, (b, e) precision, and (c, f) F_1 score of all the (a-c) batch and (d-f) online segmentation algorithms in the presence of increasing intensity noise. The test dataset was the ABO 275 μm data with added random noise. The relative noise strength was represented by the ratio of the standard deviation of the random noise amplitude to the mean fluorescence intensity. As expected, the F_1 scores of all methods decreased as the noise amplitude grew. The F_1 of SUNS was greater than the F_1's of all other methods at all noise intensities. (g-l) are in the same format of (a-f), but show the performance with the presence of increasing motion artifacts. The motion artifacts strength was represented by the standard deviation of the random movement amplitude (unit: pixels). As expected, the F_1 scores of all methods decreased as the motion artifacts became stronger. The F_1 of SUNS was greater than the F_1's of all other methods at all motion amplitudes. STNeuroNet and CaImAn batch were the most sensitive to strong motion artifacts, likely because they rely on accurate 3D spatiotemporal structures of the video. On the contrary, SUNS relied more on the 2D spatial structure, so it retained the accuracy better when spatial structures changed position over different frames.



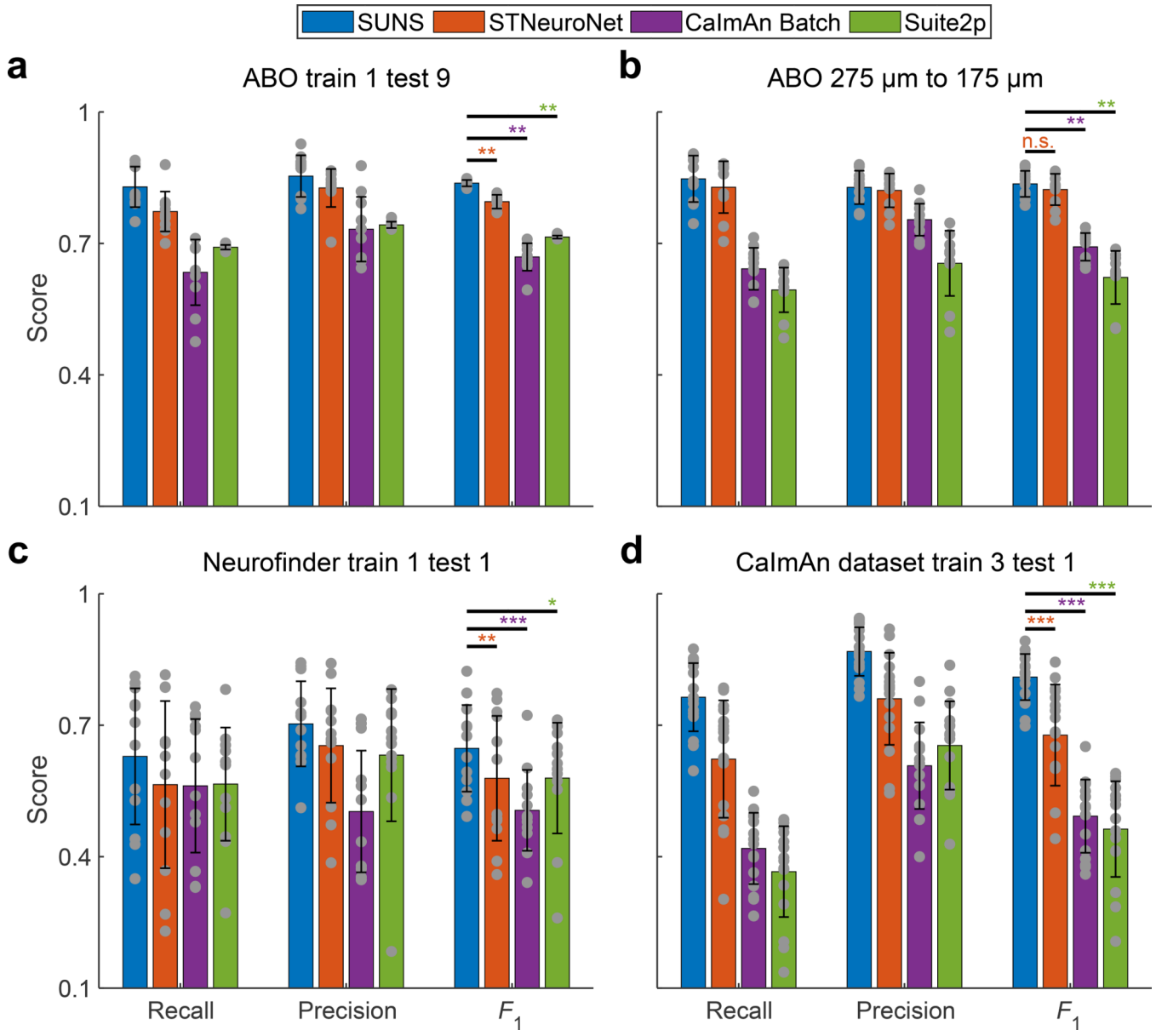
Extended Data Fig. 5. SUNS accurately mapped the spatial extent of each neuron even if the spatial footprints of neighboring cells overlapped.

SUNS segmented active neurons within each individual frame, and then accurately collected and merged the instances belonging to the same neurons. We selected two example pairs of overlapping neurons from the ABO video 539670003 identified by SUNS, and showed their traces and instances when they were activated independently. (a) The SNR images of the region surrounding the selected neurons. The left image is the maximum projection of the SNR video over the entire recording time, which shows the two neurons were active and overlapping. The right images are single-frame SNR images at two different time points, each at the peak of a fluorescence transient where only one of the two neurons was active. The segmentation of each neuron generated by SUNS is shown as a contour with a different color. The scale bar is 3 μm. (b) The temporal SNR traces of the selected neurons, matched to the colors of their contours in (a). Because the pairs of neurons overlapped, their fluorescence traces displayed substantial crosstalk. The dash markers above each trace show the active periods of each neuron determined by SUNS. The colored triangles below each trace indicate the manually-selected time of the single-frame images shown in (a). (c-d) are parallel to (a-b), but for a different overlapping neuron pair. (e) We quantified the ability to find overlapped neurons for each segmentation algorithm using the recall score. We divided the ground truth neurons in all the ABO videos into two groups: neurons without and with overlap with other neurons. We then computed the recall scores for both groups. The recall of SUNS on spatially overlapping neurons was not significantly lower (and was numerically higher) than the recall of SUNS on non-spatially overlapping neurons ($p > 0.8$, one-sided Wilcoxon rank-sum test, $n = 10$ videos; n.s.l. – not significantly lower). Therefore, the performance of SUNS on overlapped neurons was at least equally good as the performance of SUNS on non-overlapped neurons. Moreover, the recall scores of SUNS in both groups were comparable to or significantly higher than that of other methods in those groups (**$p < 0.005$, n.s. – not significant; two-sided Wilcoxon signed-rank test, $n = 10$ videos; error bars are standard deviations). The gray dots represent the scores on the test data for each round of cross-validation.



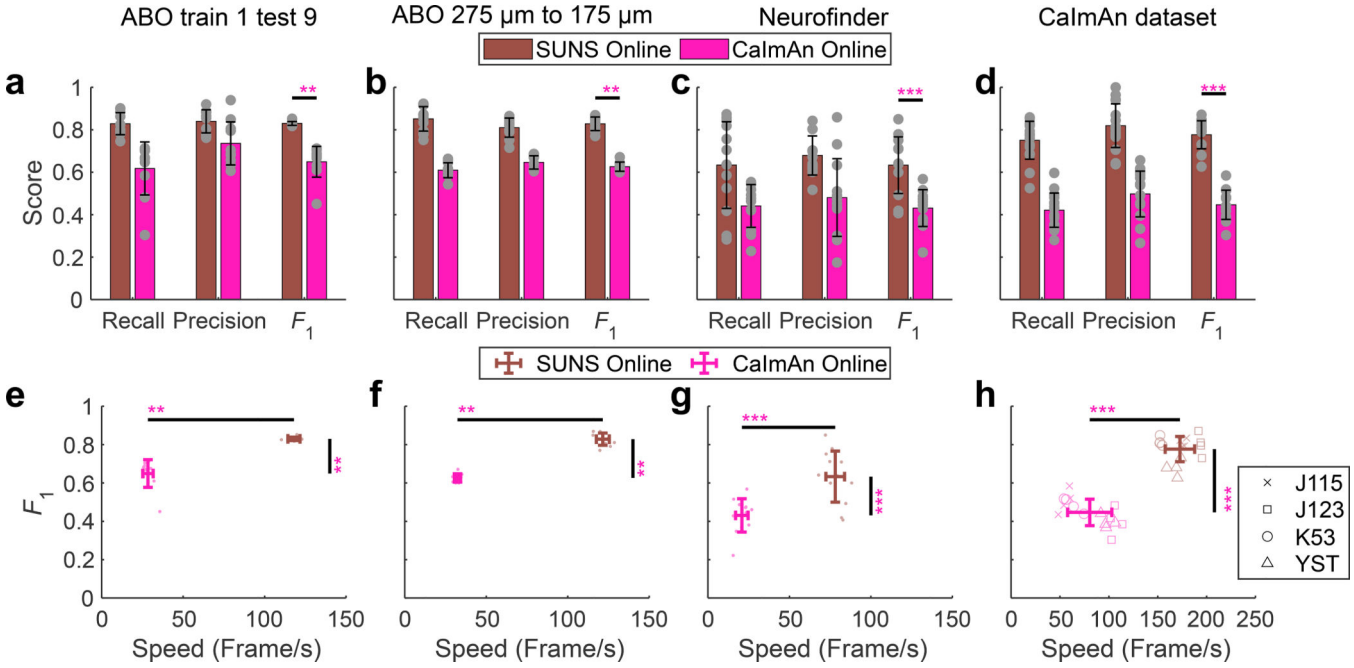
Extended Data Fig. 6. Each pre-processing step and the CNN contributed to the accuracy of SUNS at the cost of lower speed.

We evaluated the contribution of each pre-processing option (spatial filtering, temporal filtering, and SNR normalization) and the CNN option to SUNS. The reference algorithm (SUNS) used all options except spatial filtering. We compared the performance of this reference algorithm to the performance with additional spatial filtering (optional SF), without temporal filtering (no TF), without SNR normalization (no SNR), and without the CNN (no CNN) when analyzing the ABO 275 μm dataset through 10-fold leave-one-out cross-validation. (a) The recall, precision, and F_1 score of these variants. The temporal filtering, SNR normalization, and CNN each significantly contributed to the overall accuracy, but the impact of spatial filtering was not significant (*$p < 0.05$, **$p < 0.005$, n.s. - not significant; two-sided Wilcoxon signed-rank test, $n = 10$ videos; error bars are standard deviations). The gray dots represent the scores on the test data for each round of cross-validation. (b) The speed and F_1 score of these variants. Eliminating temporal filtering or the CNN significantly increased the speed, while adding spatial filtering or eliminating SNR normalization significantly lowered the speed (**$p < 0.005$; two-sided Wilcoxon signed-rank test, $n = 10$ videos; error bars are standard deviations). The light color dots represent F_1 scores and speeds for the test data for each round of cross-validation. The execution of SNR normalization was fast (~0.07 ms/frame). However, eliminating SNR normalization led to a much lower optimal th_{prob}, and thus increased the number of active pixels and decreased precision. In addition, “no SNR” had lower speed than the complete SUNS algorithm due to the increased post-processing computation workload for managing the additional active pixels and regions.



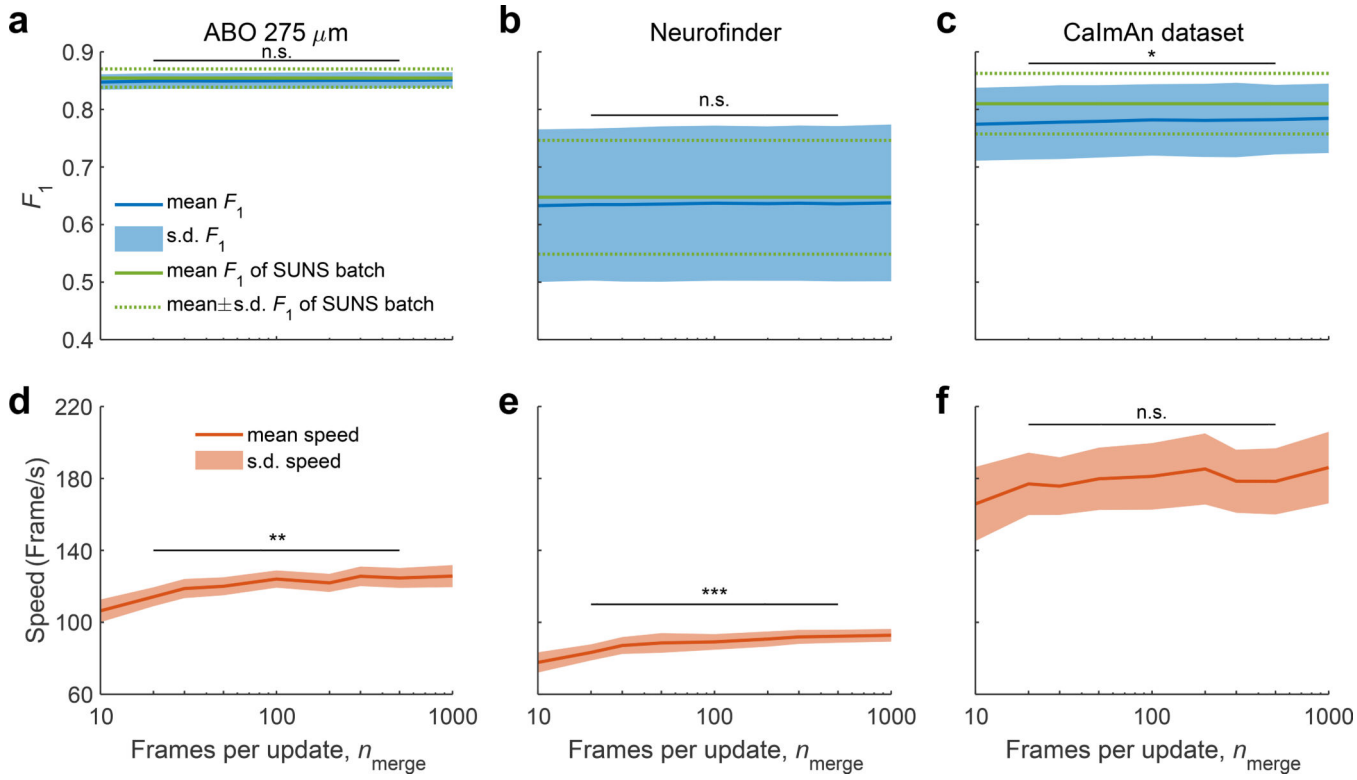
Extended Data Fig. 7. The recall, precision, and F_1 score of SUNS were superior to that of other methods on a variety of datasets.

(a) Training on one ABO 275 µm video and testing on nine ABO 275 µm videos (each data point is the average over each set of nine test videos, $n = 10$); (b) Training on ten ABO 275 µm videos and testing on ten ABO 175 µm videos ($n = 10$); (c) Training on one Neurofinder video and testing on one paired Neurofinder video ($n = 12$); (d) Training on three-quarters of one CaImAn video and testing on the remaining quarter of the same CaImAn video ($n = 16$). The F_1 scores of SUNS were mostly significantly higher than the F_1 scores of other methods (*$p < 0.05$, **$p < 0.005$, ***$p < 0.001$, n.s. - not significant; two-sided Wilcoxon signed-rank test; error bars are standard deviations). The gray dots represent the individual scores for each round of cross-validation.



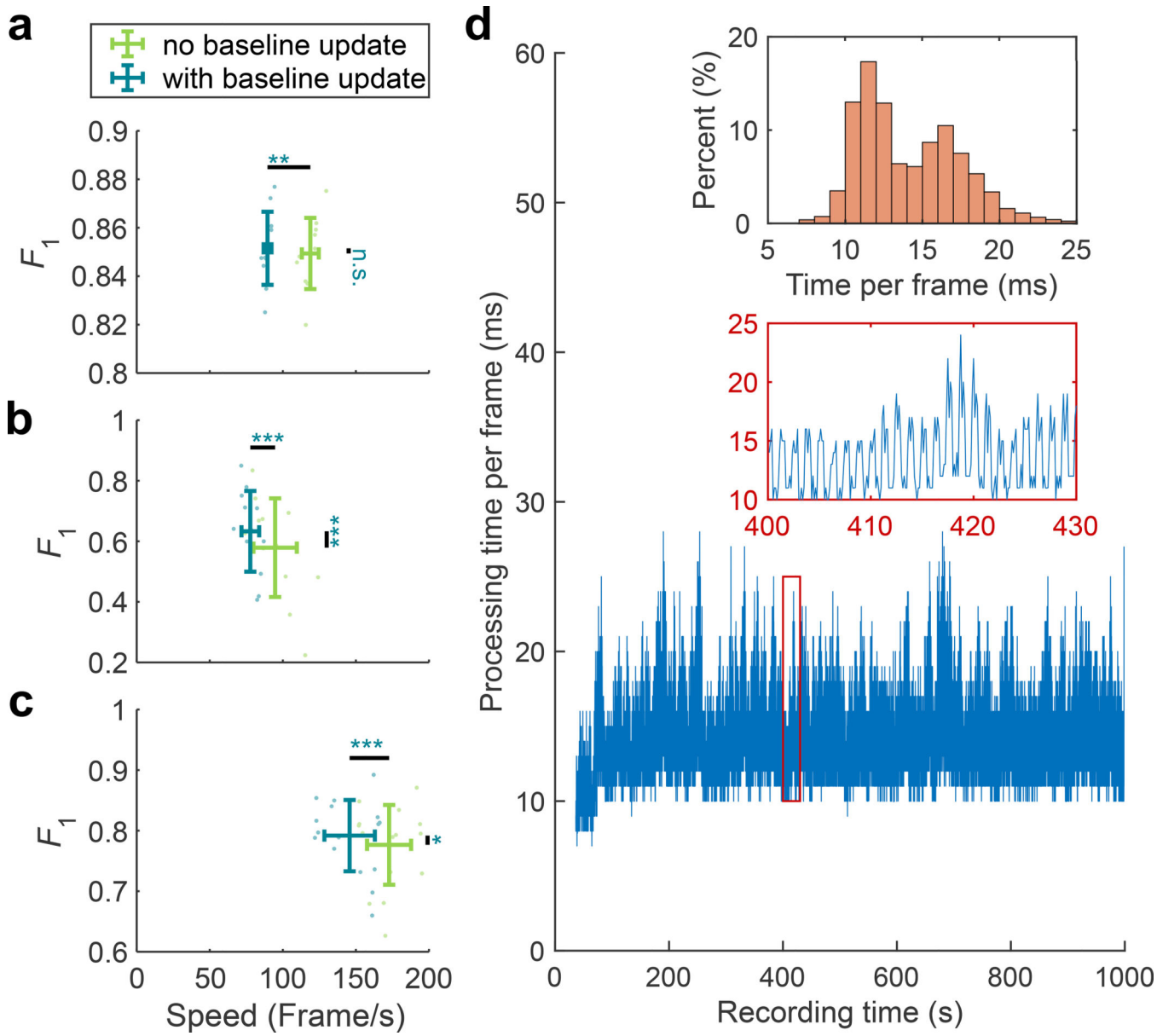
Extended Data Fig. 8. SUNS online outperformed CaImAn Online in accuracy and speed when processing a variety of datasets.

(a, e) Training on one ABO 275 µm video and testing on nine ABO 275 µm videos (each data point is the average over each set of nine test videos, $n = 10$); (b, f) Training on ten ABO 275 µm videos and testing on ten ABO 175 µm videos ($n = 10$); (c, g) Training on one Neurofinder video and testing on one paired Neurofinder video ($n = 12$); (d, h) Training on three-quarters of one CaImAn video and testing on the remaining quarter of the same CaImAn video ($n = 16$). The F_1 score and processing speed of SUNS online were significantly higher than the F_1 score and speed of CaImAn Online (**$p < 0.005$, ***$p < 0.001$; two-sided Wilcoxon signed-rank test; error bars are standard deviations). The gray dots in (a-d) represent individual scores for each round of cross-validation. The light color dots in (e-g) represent F_1 scores and speeds for the test data for each round of cross-validation. The light color markers in (h) represent F_1 scores and speeds for the test data for each round of cross-validation performed on different CaImAn videos. We updated the baseline and noise regularly after initialization for the Neurofinder dataset, but did not do so for other datasets.



Extended Data Fig. 9. Changing the frequency of updating the neuron masks modulated trade-offs between SUNS online's response time to new neurons and SUNS online's performance metrics.

The (a-c) F_1 score and (d-f) speed of SUNS online increased as the number of frames per update (n_{merge}) increased for the (a, d) ABO 275 μm, (b, e) Neurofinder, and (c, f) CaImAn datasets. The solid line is the average, and the shading is one standard deviation from the average (n = 10, 12, and 16 cross-validation iterations for the three datasets). In (a-c), the green lines show the F_1 score (*solid*) ± one standard deviation (*dashed*) of SUNS batch. The F_1 score and speed generally increased as n_{merge} increased. For example, the F_1 score and speed when using n_{merge} = 500 were higher than the F_1 score and speed when using n_{merge} = 20, and a half of the differences were significant (*p < 0.05, **p < 0.005, ***p < 0.001, n.s. - not significant; two-sided Wilcoxon signed-rank test; n = 10, 12, and 16, respectively). We updated the baseline and noise regularly after initialization for the Neurofinder dataset, but did not do so for other datasets. The n_{merge} was inversely proportional to the update frequency or the responsiveness of SUNS online to the appearance of new neurons. A trade-off exists between this responsiveness and the accuracy and speed of SUNS online. At the cost of less responsiveness, a higher n_{merge} allowed the accumulation of temporal information and improved the accuracy of neuron segmentations. Likewise, a higher n_{merge} improved the speed because it reduced the occurrence of computations for aggregating neurons.



Extended Data Fig. 10. Updating the baseline and noise after initialization increased the accuracy of SUNS online at the cost of lower speed.

We compared the F_1 score and speed of SUNS online with or without baseline and noise update for the (a) ABO 275 μm, (b) Neurofinder, and (c) CaImAn datasets. The F_1 scores with baseline and noise update were generally higher, but the speeds were slower (*$p < 0.05$, **$p < 0.005$, ***$p < 0.001$, n.s. - not significant; two-sided Wilcoxon signed-rank test; error bars are standard deviations). The light color dots represent F_1 scores and speeds for the test data for each round of cross-validation. The improvement in the F_1 score was larger as the baseline fluctuation becomes more significant. (d) Example processing time per frame of SUNS online with baseline and noise update on Neurofinder video 02.00. The lower inset zooms in on the data from the red box. The upper inset is the distribution of processing time per frame. The processing time per frame was consistently faster than the microscope

recording rate (125 ms/frame). The first few frames after initialization were faster than the following frames, because the baseline and noise update was not performed in these frames.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgements

We acknowledge support from the BRAIN Initiative (NIH 1UF1-NS107678, NSF 3332147), the NIH New Innovator Program (1DP2-NS111505), the Beckman Young Investigator Program, the Sloan Fellowship, and the Vallee Young Investigator Program received by Y.G.. We acknowledge Zhijing Zhu for early characterization of the SUNS.

References

1. Akerboom J, et al., Genetically encoded calcium indicators for multi-color neural activity imaging and combination with optogenetics. Frontiers in Molecular Neuroscience (2013).
2. Chen T-W, et al., Ultrasensitive fluorescent proteins for imaging neuronal activity. Nature. 499 (7458), 295–300, (2013). [PubMed: 23868258]
3. Dana H, et al., High-performance calcium sensors for imaging activity in neuronal populations and microcompartments. Nature Methods. 16 (7), 649–657, (2019). [PubMed: 31209382]
4. Helmchen F and Denk W, Deep tissue two-photon microscopy. Nature Methods. 2 (12), 932–940, (2005). [PubMed: 16299478]
5. Stringer C, et al., Spontaneous behaviors drive multidimensional, brainwide activity. Science. 364 (6437), eaav7893, (2019).
6. Grewe BF, et al., High-speed in vivo calcium imaging reveals neuronal network activity with near-millisecond precision. Nature Methods. 7 (5), 399–405, (2010). [PubMed: 20400966]
7. Soltanian-Zadeh S, et al., Fast and robust active neuron segmentation in two-photon calcium imaging using spatiotemporal deep learning. Proceedings of the National Academy of Sciences. 116 (17), 8554–8563, (2019).
8. Pnevmatikakis EA, Analysis pipelines for calcium imaging data. Current Opinion in Neurobiology. 5515–21, (2019). [PubMed: 30529147]
9. Klibisz A, et al.Fast, simple calcium imaging segmentation with fully convolutional networks. in Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support. 2017. Cham: Springer International Publishing.
10. Gao SConv2D. 2016; Available from: https://github.com/iamshang1/Projects/tree/master/Advanced_ML/Neuron_Detection.
11. Shen SP, et al., Automatic cell segmentation by adaptive thresholding (ACSAT) for large-scale calcium imaging datasets. eneuro. 5 (5), ENEURO.0056–18.2018, (2018).
12. Spaen Q, et al., HNCcorr: a novel combinatorial approach for cell identification in calcium-imaging movies. eneuro. 6 (2), ENEURO.0304–18.2019, (2019).
13. Kirschbaum E, Bailoni A, and Hamprecht FA, DISCo for the CIA: Deep learning, instance segmentation, and correlations for calcium imaging analysis. Preprint at https://arxiv.org/abs/1908.07957v4, (2020).
14. Apthorpe NJ, et al., Automatic Neuron Detection in Calcium Imaging Data Using Convolutional Networks, in Advances in Neural Information Processing Systems29, (Neural Information Processing Systems (Nips): La Jolla2016).
15. Mukamel EA, Nimmerjahn A, and Schnitzer MJ, Automated analysis of cellular signals from large-scale calcium imaging data. Neuron. 63 (6), 747–760, (2009). [PubMed: 19778505]
16. Maruyama R, et al., Detecting cells using non-negative matrix factorization on calcium imaging data. Neural Networks. 5511–19, (2014). [PubMed: 24705544]

17. Pnevmatikakis Eftychios A., et al., Simultaneous denoising, deconvolution, and demixing of calcium imaging data. Neuron. 89 (2), 285–299, (2016). [PubMed: 26774160]
18. Pachitariu M, et al., Suite2p: beyond 10,000 neurons with standard two-photon microscopy. Preprint at https://www.biorxiv.org/content/biorxiv/early/2017/07/20/061507.full.pdf, (2017).
19. Petersen A, Simon N, and Witten D, SCALPEL: Extracting neurons from calcium imaging data. Ann. Appl. Stat12 (4), 2430–2456, (2018). [PubMed: 30510612]
20. Giovannucci A, et al., CaImAn an open source tool for scalable calcium imaging data analysis. Elife. 8 (2019).
21. Sitaram R, et al., Closed-loop brain training: the science of neurofeedback. Nature Reviews Neuroscience. 18 (2), 86–100, (2017). [PubMed: 28003656]
22. Kearney MG, et al., Discrete Evaluative and Premotor Circuits Enable Vocal Learning in Songbirds. Neuron. 104 (3), 559–575.e6, (2019). [PubMed: 31447169]
23. Carrillo-Reid L, et al., Controlling visually guided behavior by holographic recalling of cortical ensembles. Cell. 178 (2), 447–457.e5, (2019). [PubMed: 31257030]
24. Rickgauer JP, Deisseroth K, and Tank DW, Simultaneous cellular-resolution optical perturbation and imaging of place cell firing fields. Nature Neuroscience. 17 (12), 1816–1824, (2014). [PubMed: 25402854]
25. Packer AM, Russell LE, Dalgleish HWP, and Häusser M, Simultaneous all-optical manipulation and recording of neural circuit activity with cellular resolution in vivo. Nature Methods. 12 (2), 140–146, (2015). [PubMed: 25532138]
26. Zhang Z, et al., Closed-loop all-optical interrogation of neural circuits in vivo. Nature Methods. 15 (12), 1037–1040, (2018). [PubMed: 30420686]
27. Giovannucci A, et al., OnACID: Online analysis of calcium imaging data in real time. Preprint at https://www.biorxiv.org/content/biorxiv/early/2017/10/02/193383.full.pdf, (2017).
28. Wilt Brian A., Fitzgerald James E., and Schnitzer Mark J., Photon shot noise limits on optical detection of neuronal spikes and estimation of spike timing. Biophysical Journal. 104 (1), 51–62, (2013). [PubMed: 23332058]
29. Jiang R and Crookes D, Shallow unorganized neural networks using smart neuron model for visual perception. IEEE Access. 7 152701–152714, (2019).
30. Ba J and Caruana R Do deep nets really need to be deep? in Advances in neural information processing systems. 2014.
31. Lei F, Liu X, Dai Q, and Ling BW-K, Shallow convolutional neural network for image classification. SN Applied Sciences. 2 (1), 97, (2019).
32. Yu S, et al., A shallow convolutional neural network for blind image sharpness assessment. PLOS ONE. 12 (5), e0176632, (2017). [PubMed: 28459832]
33. Ronneberger O, Fischer P, and Brox TU-Net: convolutional networks for biomedical image segmentation. 2015. Cham: Springer International Publishing.
34. Neurofinder challenge. Available from: http://neurofinder.codeneuro.org/.
35. Arac A, et al., DeepBehavior: A Deep Learning Toolbox for Automated Analysis of Animal and Human Behavior Imaging Data. Frontiers in Systems Neuroscience. 13 (20), (2019).
36. Shen D, Wu G, and Suk H-I, Deep learning in medical image analysis. Annual Review of Biomedical Engineering. 19 (1), 221–248, (2017).
37. Zhou P, et al., Efficient and accurate extraction of in vivo calcium signals from microendoscopic video data. Elife. 7e28728, (2018). [PubMed: 29469809]
38. Meyer F, Topographic distance and watershed lines. Signal Processing. 38 (1), 113–125, (1994).
39. Pnevmatikakis EA and Giovannucci A, NoRMCorre: An online algorithm for piecewise rigid motion correction of calcium imaging data. Journal of Neuroscience Methods. 291 83–94, (2017). [PubMed: 28782629]
40. Keemink SW, et al., FISSA: A neuropil decontamination toolbox for calcium imaging signals. Scientific Reports. 8 (1), 3493, (2018). [PubMed: 29472547]
41. Mitani A and Komiyama T, Real-Time Processing of Two-Photon Calcium Imaging Data Including Lateral Motion Artifact Correction. Frontiers in Neuroinformatics. 12 (98), (2018).

42. Frankle J and Carbin M, The lottery ticket hypothesis: Finding sparse, trainable neural networks. Preprint at https://arxiv.org/abs/1803.03635, (2018).
43. Yang W and Lihong X Lightweight compressed depth neural network for tomato disease diagnosis. in Proc.SPIE. 2020.

Online References

44. Oppenheim A, Schafer R, and Stockham T, Nonlinear filtering of multiplied and convolved signals. IEEE Transactions on Audio and Electroacoustics. 16 (3), 437–466, (1968).
45. Szymanska AF, et al., Accurate detection of low signal-to-noise ratio neuronal calcium transient waves using a matched filter. Journal of Neuroscience Methods. 2591–12, (2016). [PubMed: 26561771]
46. Milletari F, Navab N, and Ahmadi SV-Net: Fully convolutional neural networks for volumetric medical image segmentation. in 2016 Fourth International Conference on 3D Vision (3DV). 2016.
47. Lin T-Y, et al.Focal loss for dense object detection. in Proceedings of the IEEE international conference on computer vision. 2017.
48. Allen Brain Observatory. Available from: http://observatory.brain-map.org/visualcoding.
49. Gilman JP, Medalla M, and Luebke JI, Area-specific features of pyramidal neurons—a comparative study in mouse and rhesus monkey. Cerebral Cortex. 27 (3), 2078–2094, (2016).
50. Ballesteros-Yáñez I, et al., Alterations of cortical pyramidal neurons in mice lacking high-affinity nicotinic receptors. Proceedings of the National Academy of Sciences. 107 (25), 11567–11572, (2010).
51. YijunBao. YijunBao/Shallow-UNet-Neuron-Segmentation_SUNS. (2021). doi:10.5281/zenodo.4638171
52. YijunBao. YijunBao/SUNS_paper_reproduction. (2021). doi:10.5281/zenodo.4638135

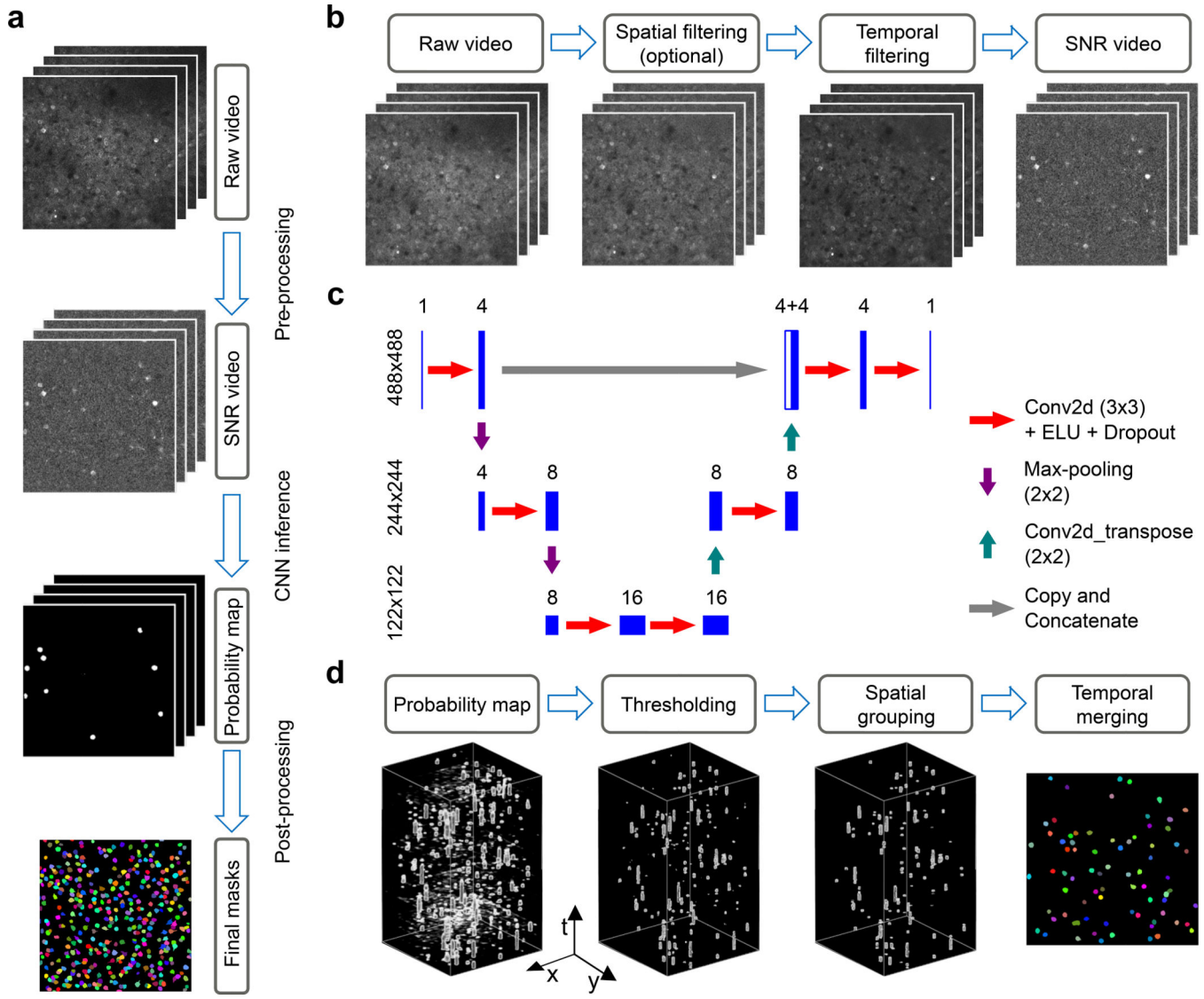


Figure 1: Schematic for the proposed fast neuron segmentation algorithm based on a shallow U-Net.

(a) The overall structure of SUNS included (b) pre-processing, (c) CNN inference, and (d) post-processing. The input is a raw registered video, and the output is a set of binary masks representing segmented neurons. (b) The pre-processing procedure used optional spatial filtering to remove large-scale background fluctuations, temporal filtering to enhance calcium transients, and SNR normalization to remove inactive neurons. The output of the pre-processing was an SNR video used for CNN inference. (c) Our CNN employed a shallow U-Net. Example dimensions of each feature map are at the left of each row. The numbers of channels in each feature map are on top of each feature. The arrows denote different local tensor operations. The output of the CNN was a probability map used in post-processing. (d) The post-processing procedure screened for active pixels, spatially grouped active pixels in each frame into active connected regions, and temporally merged active regions belonging to the same neuron across all frames.

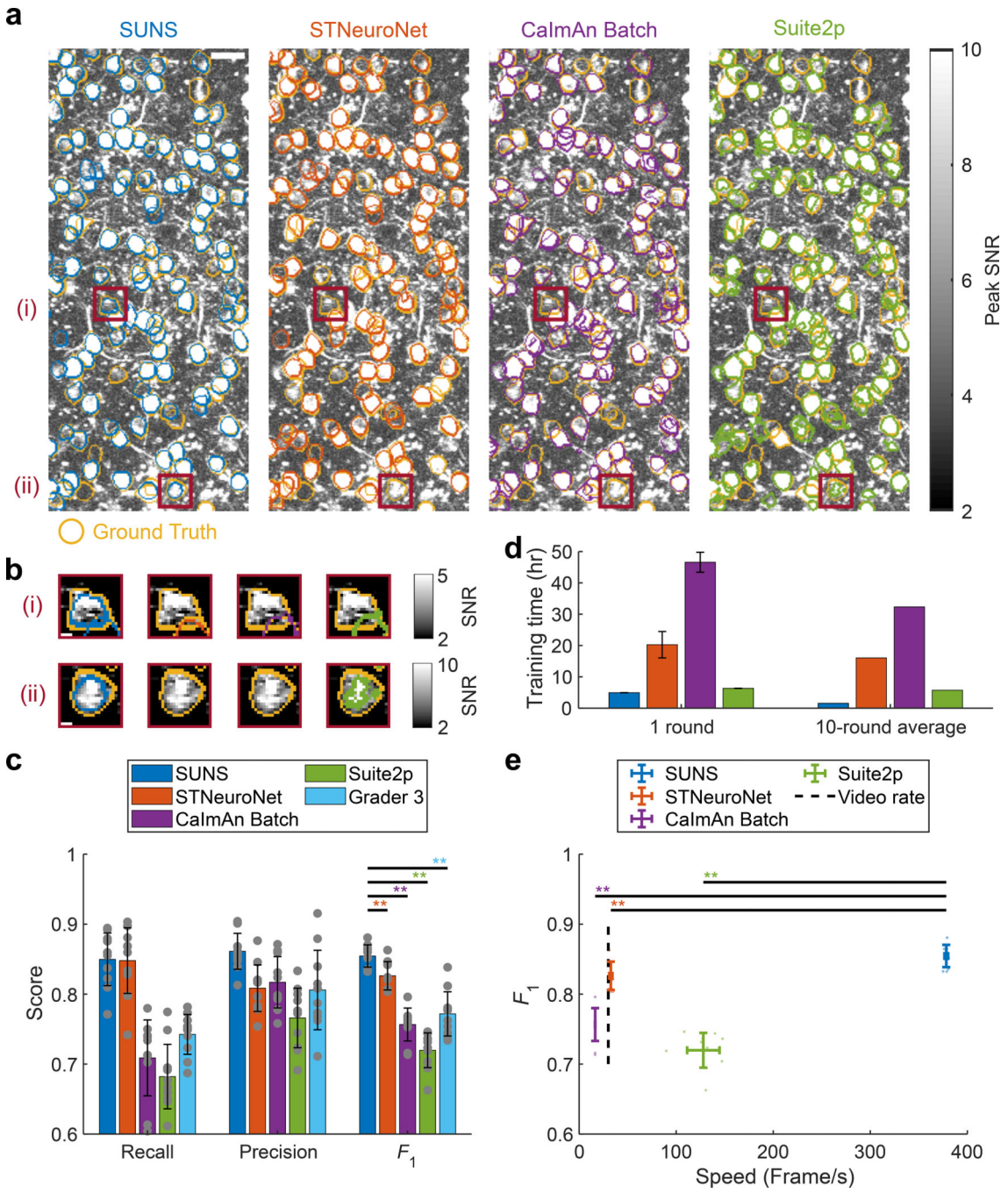


Figure 2: SUNS outperformed existing neuron segmentation algorithms in accuracy and speed on the ABO dataset.

(a) Example segmentations from a part of ABO video 524691284 for SUNS, STNeuroNet, CaImAn Batch, and Suite2p, overlaid on top of the imaging data. The grayscale image is the projection of the maximum pixel-wise SNR (Scale bar: 20 μm). The light orange outlines denote the GT neurons, and the other colors denote the neurons found by the algorithms. (b) Example neurons zoomed from the boxed regions in (a) that were identified correctly by SUNS but missed by the other methods. The images are the average SNR images around the peaks of one calcium transient from each of the pictured neurons (Scale bar: 3 μm;

Supplementary Fig. 1). (c) The recall, precision, and F_1 score of SUNS during a 10-round cross-validation were superior to that of the other methods and the independent human Grader 3 (**$p < 0.005$, two-sided Wilcoxon signed-rank test, $n = 10$ videos; error bars are standard deviations). The gray dots represent scores for the test data for each round of cross-validation. (d) SUNS required less training time than the other methods over both a single cross-validation round and a 10-round average. (e) In addition to superior detection accuracy, SUNS had faster processing speed than the other methods and the video rate (**$p < 0.005$; two-sided Wilcoxon signed-rank test, $n = 10$ videos; error bars are standard deviations). The light color dots represent F_1 scores and speeds for the test data for each round of cross-validation.

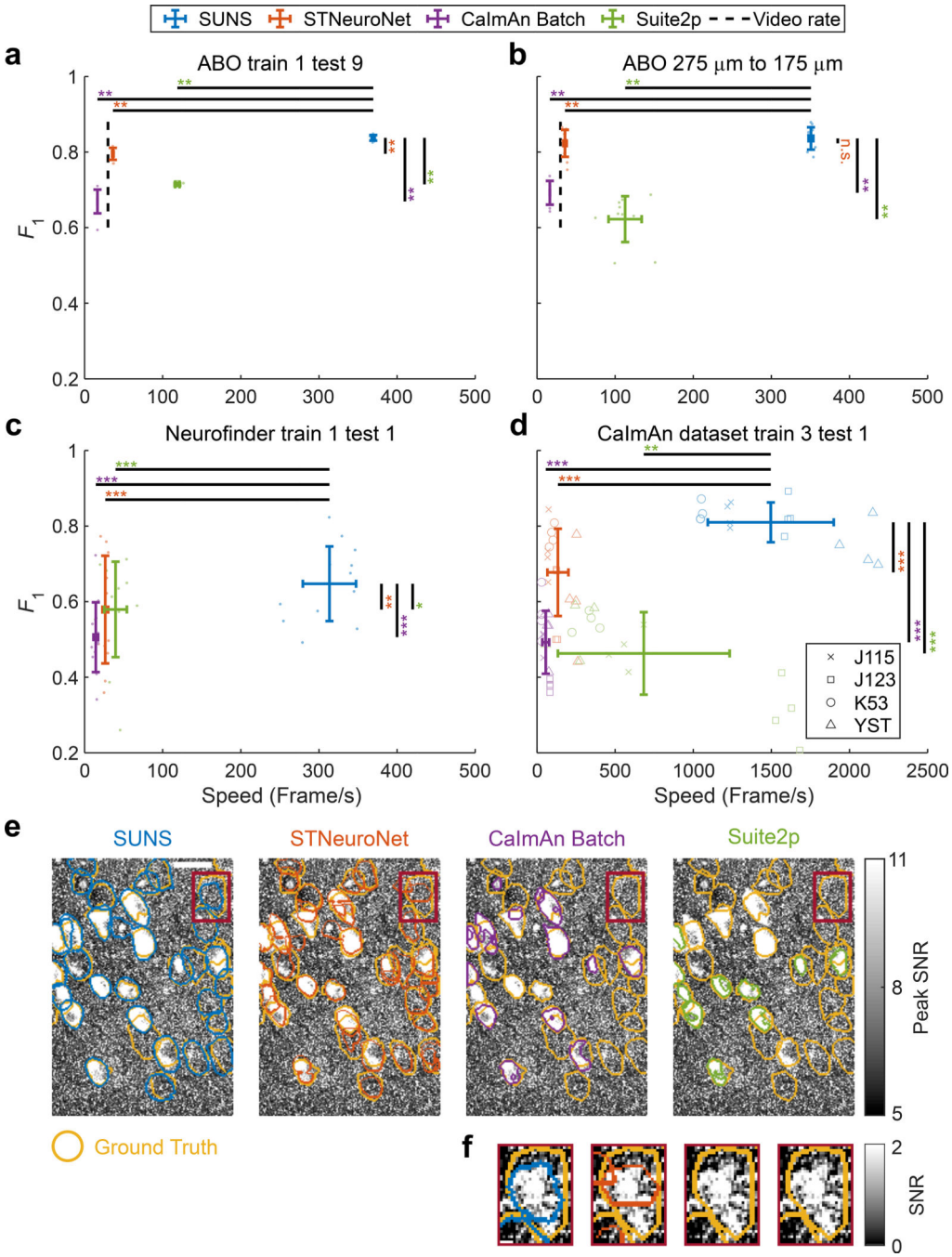


Figure 3: SUNS outperformed existing neuron segmentation algorithms in accuracy and speed when processing a variety of datasets.

(a-d) The F_1 score and processing speed of SUNS were better than STNeuroNet, CaImAn Batch, and Suite2p on four tests of generalization: (a) training on one ABO 275 μm video and testing on nine ABO 275 μm videos (each data point is the average over each set of nine test videos, $n = 10$); (b) training on ten ABO 275 μm videos and testing on ten ABO 175 μm videos ($n = 10$); (c) training on one Neurofinder video and testing on one paired Neurofinder video ($n = 12$); (d) training on three-quarters of one CaImAn video and testing on the remaining quarter of the same CaImAn video ($n = 16$) (for F_1 score and processing speed,

*$p < 0.05$, **$p < 0.005$, ***$p < 0.001$, n.s. - not significant; two-sided Wilcoxon signed-rank test; error bars are standard deviations). The light color dots in (a-c) represent F_1 scores and speeds for the test data for each round of cross-validation. The light color markers in (d) represent F_1 scores and speeds for the test data for each round of cross-validation performed on different CaImAn videos. (e) Example segmentations from the fourth quadrant of the CaImAn video J123 compare the results of all four segmentation methods, overlaid on top of the imaging data. The grayscale image is the projection of the maximum pixel-wise SNR (Scale bar: 20 μm). The light orange outlines denote the GT neurons, and the other colors denote the neurons found by the algorithms. (f) The example neuron zoomed from the boxed region in (e) that was identified correctly by SUNS but missed by the other methods. The image was the averaged SNR images around the peak of a calcium transient of the neuron (Scale bar: 3 μm; Supplementary Fig. 1).

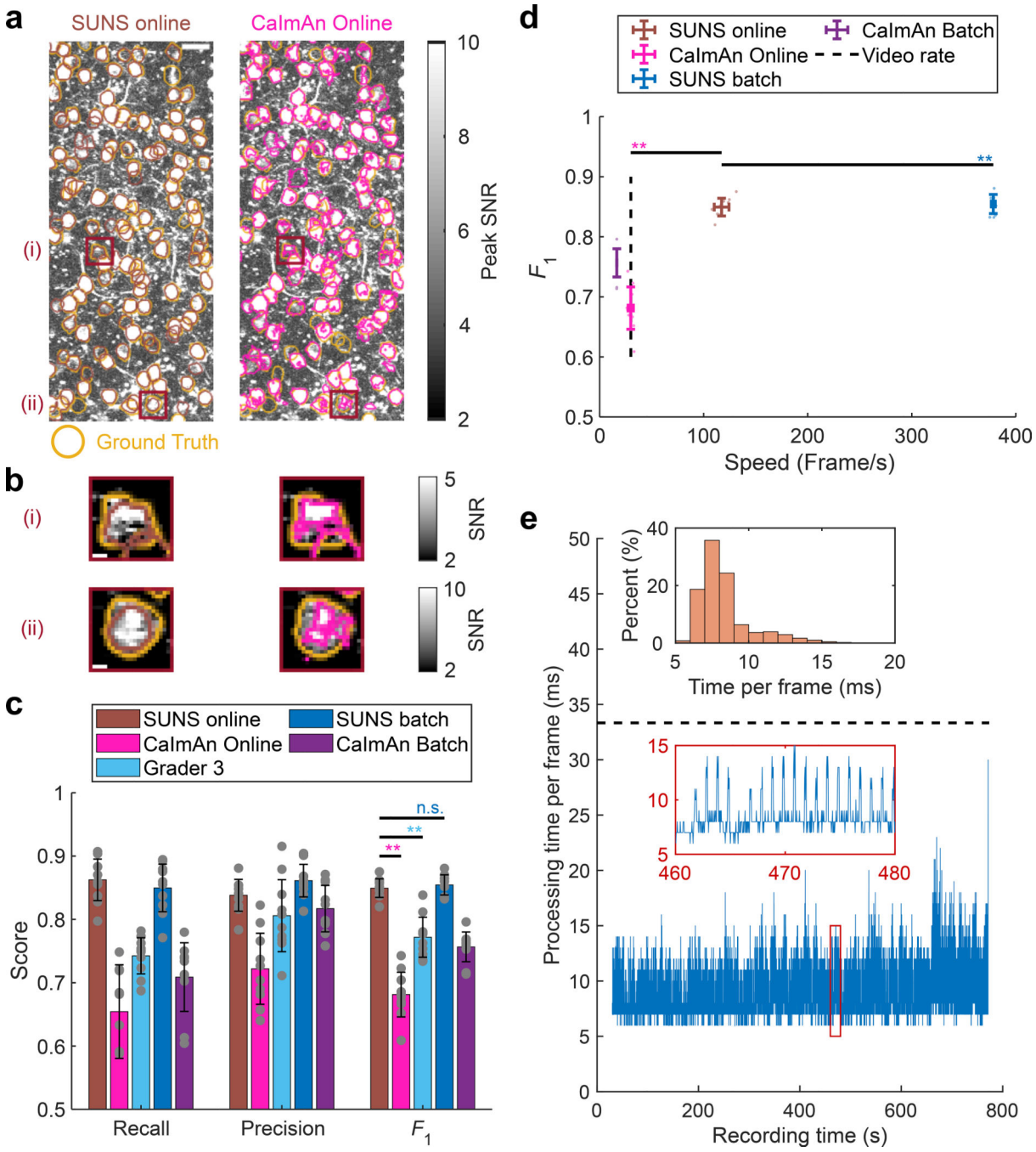


Figure 4: SUNS online outperformed CaImAn Online in accuracy and speed on the ABO dataset.

(a) Example segmentations from ABO video 524691284 compare the results of SUNS online to the results of CaImAn Online, overlaid on top of the imaging data. The grayscale image is the projection of the maximum pixel-wise SNR (Scale bar: 20 μm). The light orange outlines denote the GT neurons, and the other colors denote the neurons found by the algorithms. (b) Example neurons zoomed from the boxed regions in (a), the same region as in Fig. 2b. The images were the average SNR images around the peaks of one calcium transient from each of the pictured neurons (Scale bar: 3 μm; Supplementary Fig. 1). (c) The

F_1 scores of SUNS online during a 10-round cross-validation were superior to the F_1 scores of CaImAn Online and the independent human Grader 3, and was close to SUNS batch (**$p < 0.005$, n.s. - not significant; two-sided Wilcoxon signed-rank test, $n = 10$ videos; error bars are standard deviations). The gray dots represent the scores on the test data for each round of cross-validation. (d) In addition to superior detection accuracy, SUNS online was also faster than CaImAn Online, although slower than SUNS batch (**$p < 0.005$, two-sided Wilcoxon signed-rank test, $n = 10$ videos; error bars are standard deviations). The light color dots represent the F_1 scores and speeds for the test data for each round of cross-validation. (e) Example processing time per frame when applying SUNS online on video 501574836. The black dashed line is the microscope recording rate. The lower inset zooms in on the data from the red box. The upper inset is the distribution of processing time per frame.