Journal of Neuroscience Methods 309 (2018) 25–34

Contents lists available at ScienceDirect

Journal of Neuroscience Methods

journal homepage: www.elsevier.com/locate/jneumeth

Automated dendritic spine detection using convolutional neural networks on maximum intensity projected microscopic volumes

Xuerong Xiao[a,*], Maja Djurisic[b], Assaf Hoogi[c], Richard W. Sapp[b], Carla J. Shatz[b], Daniel L. Rubin[c]

[a] Department of Electrical Engineering, Stanford University, David Packard Building, 350 Serra Mall, Stanford, CA 94305, USA
[b] Departments of Biology and Neurobiology, and Bio-X, Stanford University, James H. Clark Center, 318 Campus Dr, Stanford, CA 94305, USA
[c] Department of Biomedical Data Science, Stanford University, Medical School Office Building, 1265 Welch Rd, Stanford, CA 94305, USA

ARTICLE INFO

Keywords:
Dendritic spine detection
Deep learning
Convolutional neural networks

ABSTRACT

Background: Dendritic spines are structural correlates of excitatory synapses in the brain. Their density and structure are shaped by experience, pointing to their role in memory encoding. Dendritic spine imaging, followed by manual analysis, is a primary way to study spines. However, an approach that analyses dendritic spines images in an automated and unbiased manner is needed to fully capture how spines change with normal experience, as well as in disease.

New method: We propose an approach based on fully convolutional neural networks (FCNs) to detect dendritic spines in two-dimensional maximum-intensity projected images from confocal fluorescent micrographs. We experiment on both fractionally strided convolution and efficient sub-pixel convolutions. Dendritic spines far from the dendritic shaft are pruned by extraction of the shaft to reduce false positives. Performance of the proposed method is evaluated by comparing predicted spine positions to those manually marked by experts.

Results: The averaged distance between predicted and manually annotated spines is 2.81 ± 2.63 pixels (0.082 ± 0.076 microns) and 2.87 ± 2.33 pixels (0.084 ± 0.068 microns) based on two different experts. FCN-based detection achieves F scores > 0.80 for both sets of expert annotations.

Comparison with existing methods: Our method significantly outperforms two well-known software, NeuronStudio and Neurolucida (p-value < 0.02).

Conclusions: FCN architectures used in this work allow for automated dendritic spine detection. Superior outcomes are possible even with small training data-sets. The proposed method may generalize to other datasets on larger scales.

1. Introduction

Dendritic spines are postsynaptic structures emanating from dendritic branches of most excitatory neurons in the mammalian brain. Presynaptic nerve terminals appose functional spines, and together they form excitatory synapses, units of excitatory chemical synaptic transmission in the brain. A mature dendritic spine consists of a spine head connected to the dendritic branch by a thin neck. A spine head houses postsynaptic molecular machinery needed for generating excitatory postsynaptic potentials, but also to initiate growth or retraction of the spine structure itself. With the advent of modern high-resolution imaging techniques, it has been shown that the density of dendritic spines, and thus the overall density of synaptic contacts, changes dynamically with experience during development and healthy adulthood (Djurisic et al., 2013; Fu et al., 2012; Majewska and Sur, 2003; Zuo et al., 2005a, b). In addition, under pathological conditions, such as Alzheimer's, Huntington's or Parkinson's diseases, permanent loss of spines and synapses has been extensively documented, and correlated with cognitive decline or loss of motor function (Day et al., 2006; Graveland et al., 1985; Grutzendler et al., 2007; Shankar et al., 2007).

Modern high-resolution microscopy techniques generate vast amounts of dendritic spine images. However, there is a bottleneck to in-depth analysis of the spine data due to manual inspection of large datasets, which is laborious and inconsistent among different readers. Moreover, dendritic spine structure itself is currently described by one of the three categories that represent most obvious spine forms – mushroom, thin, or stubby; while this approach is necessitated by manual categorization process, it is likely simplifying the continuum of change that these synaptic structures undergo with plasticity. Therefore, generalizable and automated data-driven detection

* Corresponding author.
E-mail address: xuerong@stanford.edu (X. Xiao).

https://doi.org/10.1016/j.jneumeth.2018.08.019
Received 7 February 2018; Received in revised form 16 August 2018; Accepted 16 August 2018
Available online 18 August 2018

procedure to supplant manual identification of dendritic spines is needed.

A number of strategies have been employed so far for automated spine detection. Most of them rely on extraction of the dendritic shaft as a crucial step in the detection pipeline. For example, Yuan et al. (2009) used a grayscale skeletonization algorithm that is constrained with the minimum description length principle and other neuroanatomy-specific constraints. Zhang et al. (2007) traced the centerline of dendrite and spines to extract the backbone using curvilinear structure detectors. Another approach was to extract dendritic shaft and segment the spines according to width-based criteria that utilize a common morphological feature of the spines (Bai et al., 2007). In Janoos et al. (2009), dendritic skeleton and spines were identified using a medial geodesic function after dendrites were reconstructed using surface representation; this algorithm is sensitive to parameters in image acquisition and to the type of neurons being analysed. Based on traced dendritic shaft, Mukai et al. (2011) performed spine detection using eigenvalue images. Su et al. (2014) proposed an iterative backbone extraction method, directional morphological filters and shortest path techniques to extract boundaries of dendrite.

Another set of approaches studies the geometry of dendrites and dendritic spines to build mathematical models to help with detection and segmentation (Blumer et al., 2015; Fan et al., 2009; He et al., 2012; Zhang et al., 2010). Fan et al. (2009) used curvilinear structure detectors and an adaptive level-set model to detect spines. Further, Zhang et al. (2010) improved on their previous detection results by using gradient vector flow method. He et al. (2012) proposed more accurate detection using the minimal cross-sectional curvature of spine tips; a region-growing method was then employed to extract entire spines. Blumer et al. (2015) developed a statistical dendrite intensity and a spine probability model; their approach obviated a need for manual annotations to create a reference standard by generating synthetic fluorescent images from automated scanning electron microscope data; however, the detection in their method was biased towards the shapes of spines that were most frequent in the training set. Lastly, some efforts used extracted spine features to further distinguish spines from non-spine objects (Yuan et al., 2009; Zhang et al., 2007). In most of these methods, there is a need to manually adjust parameters in different steps of the algorithm. Furthermore, performance tends not to be consistent across different datasets, or the algorithms are designed to efficiently detect certain spine categories.

An existing software package often used for dendritic spine detection is Neurolucida 360 (Dickstein et al., 2016). Its spine detection depends both on dendritic shaft tracing and modelling of the protrusions from the dendritic shaft using meshes. The software allows for both automatic and user-guided dendritic shaft tracing. Another software package that is frequently used is NeuronStudio (Rodriguez et al., 2008). For spine detection, NeuronStudio uses candidate voxel clustering based on local intensity threshold, distance to the closest point on the surface, and expected maximum spine height (Rodriguez et al., 2008). Both NeuronStudio and Neurolucida 360 have a semi-automated approach where seed-pixels and input parameters are used to trace dendritic shaft; on our dataset, this semi-automated approach works better than their fully automated versions.

Recently, deep convolutional neural networks (CNNs) have demonstrated enormous potential in the field of medical image analysis (Christ et al., 2016; Kamnitsas et al., 2017). Unlike traditional machine learning techniques, deep neural networks do not require hand-crafted features and can be trained end-to-end for object detection and semantic segmentation. Unlike patch-wise classification in traditional computer vision, CNNs can be engineered to capture contextual information to reduce false positive rates. A subcategory of CNNs, fully convolutional networks (FCNs), allows input images to have varied sizes and produce correspondingly sized output image with efficient inference and learning. FCNs have become the key component in semantic segmentation systems (Long et al., 2015). The original FCNs perform learnable up-sampling using fractionally strided convolution. In the case of low-resolution image input, FCNs can up-sample with efficient subpixel convolutions to be able to generate a super-resolution image output (Shi et al., 2016); this process can also be applied to an image style transfer task where the content of one image is transformed by applying the style from another (Johnson et al., 2016). In biomedical images, recent successes in precise image segmentation were achieved by U-Net architecture (Ronneberger et al., 2015). In U-Net, contextual information is propagated to up-sampling layers by concatenating output of lower layers to higher layers, providing more feature channels.

Here, we apply FCNs to automated dendritic spine detection. Similar to U-Net, we add the output of lower layers to higher layers to combine low-level features with global features. This paper has three main contributions. First, to the best of our knowledge, this is the first instance of FCNs applied to automated dendritic spine detection, with the exploration of different up-sampling methods. FCNs automatically extract features for spine detection. Second, the pipeline with FCN is self-contained: once the FCN model is trained, there is a minimal requirement for manual parameter tuning; the detection result is not highly dependent on the post-processing step that extracts dendritic shaft. Third, this method outperforms existing available software packages for spine detection overall, with higher success rate on thin and faint spines that are of neurobiological significance. Although training deep networks requires large datasets, our method works on a small number of images. The method should be generalizable and robust, and thus we expect it to be deployed on a different dataset or a much larger dataset with more significant diversity.

2. Material and methods

2.1. Dendritic spine detection pipeline

Our approach for performing automated spine detection and evaluating the results is summarized in Fig. 1 and is detailed in the accompanying sections below. First, dendritic spines are imaged as high-resolution 3D volumes (“raw image”). After deconvolution, several maximal intensity projection (MIP) images are generated from each volume, ensuring that all the imaged spines are included. These projected 2D images are input into the CNNs. The output of the CNNs leads to a probability map of the location of dendritic spines. Predicted positions of dendritic spines are extracted from the output of the network by binarizing the probability maps. Dendrite shaft extraction is performed to prune false detections that are far away from the dendrite shaft. Finally, to evaluate accuracy of spine detections, the predicted positions of the spines are compared with those that were annotated by two expert readers.

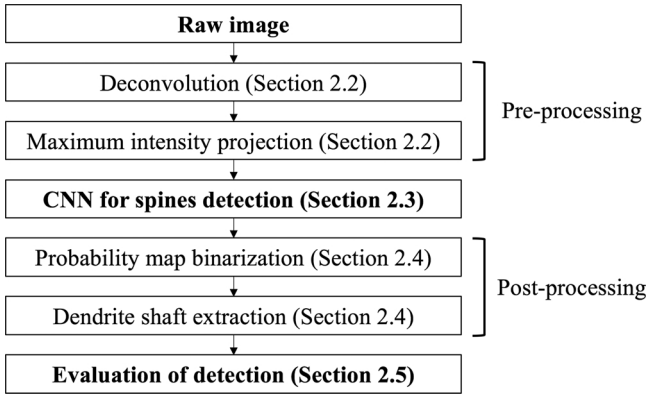

Fig. 1. Pipeline of dendritic spine detection starting from raw image, including pre-processing and post-processing steps.

2.2. Dendritic spine imaging and pre-processing

Dendritic spine imaging was performed on visual cortex tissue obtained from a transgenic YFP-H mouse line, in which expression of yellow fluorescent protein (YFP) was localized to cortical layer 5 (L5) (Feng et al., 2000). A cohort of ten wild-type (WT) male mice, postnatal day 90 (P90) or older, a part of a previous larger study (Djurisic et al., 2013) were used here. Imaging was as described previously (Djurisic et al., 2013). Briefly, distal apical dendrites of L5 neurons were imaged with a Leica TCS SP2 AOBS confocal microscope (Leica Microsystems Inc, Bannockburn, IL) using a 63X oil immersion objective and 8X zoom. Images were acquired at 1024 × 1024 pixels in x- and y-axis (0.03 µm per pixel) and at 0.16-0.2 µm z-axis resolution. Apical tufts and the primary dendritic shaft immediately below the main branching of the dendrite (approximately layer 2/3 border) were used in the analysis.

For this project, there are 20 volumes of microscopic images with a total of 962 optical slices. Each of these volumes contains different number of slices, varying from 31 to 88. Each slice was de-convolved in Huygens Software (Scientific Volume Imaging, The Netherlands) with target signal-to-noise ratio of 35. De-convolved images within a stack were grouped into sub-stacks such that maximal number of distinctive spines appeared after maximum intensity projection (MIP) in the z-axis. Fig. 2 shows a projected raw (A) and a de-convolved sub-stack (B). Slices in the beginning or end of the original volume containing no dendritic structures were discarded. The sub-stacks were collapsed to create a total of 60 MIP images, which comprised our dataset. Two neuroscience experts provided manual annotations on the MIP images independently using Adobe Illustrator (Adobe Systems), in combination with Image J software (MBF “Image J for Microscopy” collection of plugins by Tony Collins; US National Institutes of Health). The annotations consisted of one seed pixel placed in the approximate center of each spine head.

The 60 MIP images were split into training, validation, and test sets, for the purposes of learning from examples, selecting models and tuning hyper-parameters, and evaluating the performance on unseen data, respectively. The training set was randomly selected from 16 different volumes out of the 20 volumes; the validation set was from 2 randomly selected volumes out of the rest of the 4 volumes; the test set was from the remaining 2 volumes. It was ensured that the sub-stacks and thus the MIP images from the same volume were kept together within the same type of dataset. The resulting training set has 48 MIP images; the validation set contains 5 MIP images, and test set 7 MIP images.

2.3. CNN for dendritic spine detection

We designed the FCN network so that the output image has the same height and width as the input image (Long et al., 2015). Standard processes of convolution, batch normalization, nonlinear activation, and pooling operations were applied as described previously (Ioffe and Szegedy, 2015; Szegedy et al., 2015). Briefly, during convolutions, learnable kernels scan across each layer’s input, and the output is generated as the inner products between weights of each kernel and different local regions of the input image. To help regularization and address internal covariate shift, we performed batch normalization (BN) by normalizing data in each mini-batch during training. In the test phase, the normalization was according to the estimated population statistics, as described in Ioffe and Szegedy (2015). After BN, we applied leaky rectified linear units (leaky ReLU) to introduce nonlinearity into the network. After leaky ReLU, we applied max-pooling to down-sample the spatial size of the representation to reduce the amount of computation and avoid overfitting.

In addition to the standard down-sampling processes, we used a modified up-sampling procedure that chooses between fractionally strided convolutions (FSC) or efficient subpixel convolutions (ESPC), depending on the result of validation. FSC is also known as transposed convolution; it is commonly used for learnable up-sampling. Unlike FSC, the ESPC network does not calculate the multiplication of weights with inserted zeros in the input matrix and is more efficient as a result; it performs convolutions in low resolution space multiple times and interweaves the extracted feature maps to form the final high-resolution output in the last layer (Shi et al., 2016). Instead of up-sampling in the last layer (Shi et al., 2016), we used ESPC in each up-sampling step to increase the expressive power of the network.

Fully convolutional version of GoogleNet (Szegedy et al., 2015) was used, and both FSC and ESPC were studied as different ways for up-sampling, as depicted in Fig. 3. In up-sampling steps, the network can combine fine local features from low-level layers with coarse global features from high-level layers; this is referred to as a skip architecture (Long et al., 2015). For convenience, we refer to the last layer before up-sampling as the “scoring layer”, as it would have been the layer that generate scores for classification tasks with final fully connected layers. We modified the skip architecture to use layers immediately before pooling layers, as layers with fine local features, instead of previously used pooling layers (Long et al., 2015). The network takes 2D gray-scale images as an input, each of size 1024 × 1024 pixels. The output of the network is a tensor of size 1024 × 1024 × 2 with 2 channels, each corresponding to a class: “spines” or “non-spines”. The output of the network is run through a softmax layer to generate probability maps for the two classes. To unify the nomenclature of different models, we

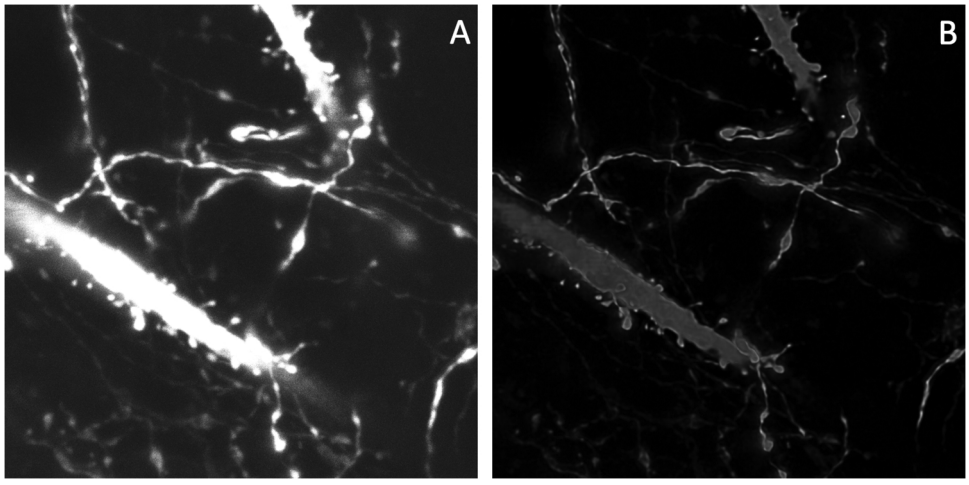

Fig. 2. Projected sub-stack from raw microscopic images (A) and de-convolved images (B).

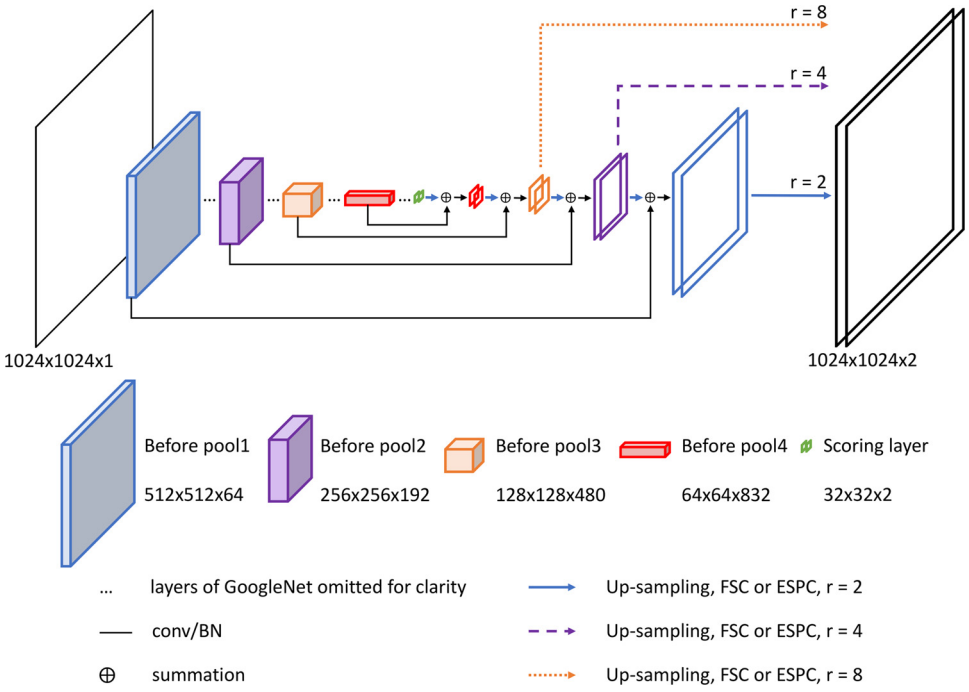

Fig. 3. Architecture of up-sampling using FSC or ESPC with r = 2, r = 4, and r = 8. Input and output of the model are in black. Intermediate layers are shown in different colors. Layers of the same color have the same height and width. Layers that are identical between GoogleNet and our architecture (blocks of conv/BN/ReLU/pooling), up to the scoring layer are not depicted. After each intermediate up-sampling step, we sum the output of each up-sampled layer with the conv/BN output of the layer immediately before the pooling operation. Only the layers involved in the summation are shown. Each up-sampling step is denoted by arrows: the difference between architectures having r = 8, r = 4, and r = 2 lies in the last up-sampling step, denoted by arrows of different styles. For r = 8, the up-sampling is directly from 128 × 128 × 2 (orange layer) to the output layer of size 1024 × 1024 × 2; for r = 4: 4-fold up-sampling from the 256 × 256 × 2 layer (purple) to the output layer; for r = 2, two-fold up-sampling from 512 × 512 × 2 (blue) to the output layer. The input image has one channel with a size of 1024 × 1024 × 1. The output layer is a tensor of shape 1024 × 1024 × 2 which is run through a softmax layer that generates probability maps for the two classes being predicted (spines or non-spine).

denote the last up-sampling fold "r" after both FSC and ESPC, e.g. FSC-8 means FSC model with the final up-sampling step of r = 8. We explored architectures having r = 8, r = 4, and r = 2.

Before training the network, we z-scored the input images using the mean and the standard deviation of the training images to limit the variation range of the weights during learning. The weights in the convolutional layers were initialized using He initialization (He et al., 2015); the weights in the transposed convolution layers (or FSC layers) were initialized using bilinear kernels (Long et al., 2015). Updates of the weights were performed iteratively using stochastic gradient descent (SGD), in which the true gradient of the loss function is approximated by the gradient against a mini-batch of samples at each step (LeCun et al., 1998). Based on our experiments during validation, the batch size was set to 2 with an initial learning rate of 0.005 that decayed 10% in every 500 epochs. The learning rate was chosen to prevent overshooting, avoid being trapped in a local minimum, or taking too long to train. In order to select among the six architectures, we saved models with different weights initialization and checkpoints during training. The number of epochs for each model was selected to achieve the best performance of each model on validation data.

Training deep networks requires a lot of data, and data augmentation is commonly performed to inflate the training data while preserving the ground truth (Yaeger et al., 1997). To alleviate overfitting, we shuffled the training data and performed the following transformations with experimentally determined probabilities to augment each mini-batch of samples during SGD: (1) with a probability of 0.7, translating the image in four directions by a distance randomly chosen within the range of [-256, 256] pixels and padding the shifted part with zeros, (2) with a probability of 0.5, flipping the image around the horizontal or vertical axis, (3) rotating the image by a random integer in multiples of 90 degrees clockwise or counter-clockwise, and (4) with a probability of 0.5, raising the overall intensity of the image to a power drawn from a Gaussian distribution with mean of 1.0 and standard deviation 0.3. The data augmentation was performed stochastically to prevent the regeneration of the same sample.

The ground truth needs to have the same shape as the output predictions from the network. Consequently, manual annotations in the form of seed pixels were converted to binary masks of size 1024 × 1024 with circular regions centred at the seed pixels. The radii of the circular regions were set to 3 pixels: using smaller radii increased training time, whereas circles with larger radii tended to enclose background pixels that are non-spine. The probability maps generated from the output of the network were compared with these binary masks to compute the loss function. Cross entropy loss between the prediction and the ground truth was computed and regularized by the L2 norm of the weights. Due to batch normalization, L2 regularization was not used as the primary method to prevent overfitting. Therefore, the coefficient was set to be a small number, 1e-6. The loss was minimized using Adam optimization (Kingma and Ba, 2014).

2.4. Post-processing

Post-processing steps are independent from the rest of the pipeline, including probability map binarization and dendritic shaft extraction, as shown in Fig. 1. We binarized the output probability maps generated from the network in order to extract predictions in the form of pixel coordinates. The threshold for the binarization is the product of a constant coefficient β with the mean of the probability map. The value of β is determined during validation and is fixed at 200 for all experiments. After the binarization, the centroid of each connected region is selected as a candidate position of a spine.

To eliminate false detections that can be located far away from the main branches of the dendrite in some images, we binarize the MIP image using Otsu's method and remove small open areas to estimate the boundaries of the main dendritic shaft. Here we make another assumption that spines are within a cut-off distance of 60 pixels (1.8 microns) away from the approximate boundaries of the shaft. This assumption is based on the typical length of spine necks. Any detection outside the cut-off is discarded. The parameters used in this step are fixed in all experiments. As will be shown in Section 3.1, this step, which was designed to improve the performance, turned out to be unnecessary for the best-performing models.

2.5. Evaluation of detection

Predicted spines that were too remote from the dendritic shaft were

pruned away and not considered in the evaluation. For a predicted spine to be counted as a true positive it had to be within a distance of δ pixels from the closest seed pixel. Unless otherwise noted, all results reported used a fixed δ at 16 pixels. The robustness of the choice of both the binarization coefficient β and the distance tolerance δ is analysed in Section 3.1. Although some spines in one MIP image may re-appear in the MIP image from a neighbouring sub-stack, we treat each MIP image independently during annotation and evaluation. We evaluated the detection pipeline on a per-spine basis, using the results on the test images as our final results. The metrics include the number of true positives (TP), false positives (FP), false negatives (FN), precision (= TP / (TP + FP)), recall (= TP / (TP + FN)), and F score (harmonic mean of precision and recall). We also calculated the mean Euclidean distance between our predictions and the manual markings.

The final results are compared with the semi-automated versions of two available software packages (Section 3.2): NeuronStudio (NS) and Neurolucida 360 (NL). Both NS and NL are based on skeleton extraction and need human intervention to achieve reasonable results. In NS, seed pixels on the dendritic shaft need to be provided manually, after which the program traces the skeleton and detect spines at the vicinity of the shaft. In NL, we used the Rayburst Crawl method to trace the main shaft, and the tracing was mostly manual for better performance. Parameters such as outer range, minimum height, detector sensitivity, and minimum voxel count were tuned to optimize the performance. Two dimensional MIP images were input to NS, and the pre-MIP sub-stacks were input to NL as three dimensional volumes. If NS failed to trace the skeleton and detected zero spine in an image, or if NL failed to load a sub-stack with only two raw image slices, the corresponding MIP image was excluded from evaluation. The coordinates of predicted spine heads from NS and NL were regarded as the final results and were compared with the manual annotations from the two readers in the same way as the evaluation of our method. Lastly, Wilcoxon signed-rank tests were conducted to calculate the p-values based on F scores per image from different methods.

All experiments related to CNNs were conducted in TensorFlow (Abadi et al., 2016). Each training of 1000 epochs took less than one day on a Tesla K40c GPU from Nvidia. Each Tesla K40c has 12 G of memory. Our model took on average 4 s to make predictions on one image. Our data and code will be available upon request.

3. Results and discussion

3.1. Model selection from different FCN architectures

The models we compared in the validation process include six architectures with different up-sampling schemes: FSC-8, FSC-4, FSC-2, ESPC-8, ESPC-4, and ESPC-2. In addition, we compared the validation results with those from GoogleNet FCN-8 s, VGG16 FCN-8 s (Long et al., 2015), and UNet (Ronneberger et al., 2015) trained on our data. The model selection was based on F scores achieved on the validation set using Reader 1′s annotations, because only Reader 1′s annotations were used in training; the performance was also evaluated based on Reader 2, as shown in Table 1. The model with ESPC-4 achieves the highest F score of 0.83 when aggregating TPs, FPs, and FNs for the entire validation dataset based on Reader 1 (or 0.83 ± 0.006 when averaging F scores for each image), as well as the highest F score of 0.83 based on Reader 2 (or 0.84 ± 0.006 when averaging F scores for each image). Wilcoxon signed-rank tests were used to test the difference in the performance between ESPC-4 vs. all other architectures (Table 3 in Supplemental Materials): while there is no statistically significant difference in recall using ESPC-4 vs. other architectures, the precision of ESPC-4 was significantly higher relative to FSC-4, ESPC-2, and GoogleNet FCN-8 s ($p < 0.05$), and trended higher against all other architectures. Therefore, we used ESPC-4 to evaluate the test set.

3.2. Post-processing: effect of shaft extraction is negligible

The effect of shaft extraction (SE) as a post-processing step is evaluated for each architecture by changes in F scores before and after the shaft extraction (Table 1). After SE, precision modestly increased for all the architectures tested, as it successfully removed false positives. On the other hand, recall values decreased during shaft removal; one possible explanation is that spines that were correctly detected in the vicinity of the shaft of low S/N (i.e., shaft is in focus in the neighbouring sub-stack) were removed in the SE step as false positives. The ESPC-4 model, along with FSC-8, VGG16 FCN-8 s, are not significantly affected by the SE step on any of the performance metrics (Table 4, Supplemental Materials). For all other architectures, the SE step modestly, but significantly improved precision, while recall and F scores were unaffected (Table 4, Supplemental Materials). Unlike previously published methods that rely on shaft extraction for accurate spine detection, our pipeline does not depend on shaft extraction for better performance.

3.3. Visualization of weights and activation maps

There are 64 filters in the first convolutional layer, each having a shape of 7×7. The trained weights in the first convolutional layer of ESPC-4 are shown in Fig. 4. It is noticeable that the weights have captured low-level features such as edges, contrast, and local patterns.

The second channels of the activation maps from selected layers are shown in Fig. 5. Those selected layers are directly involved in the summation operation in the network, fusing local (fine) features with global (coarse) features. The summation of the first two columns generates the third column. The last map shown (C3) is the closest to the output layer, and it contains high signals for regions representing spines. Also visible in C3 are few boutons (non-spine category), because of their similar morphology to spines and due to high signal value. However most of the boutons are suppressed by the probability map binarization and shaft extraction process.

3.4. Addressing overfitting

As mentioned in Section 2.2, the validation set was used for model selection and hyper-parameter tuning. We used F scores as the metric for selecting models. F scores can only be computed after binarization of the probability map. The performance evaluated before the shaft extraction (SE) during training process is shown in Fig. 6, using the training of ESPC-4 as an example. The losses and F scores on the training data and validation data versus training epochs have close values before 700 epochs, and we used early stopping during training to prevent further overfitting. Although we have a small number of images, each image has high resolution and contain around 30 spines. Therefore, our training images contain around 1000 spine incidences. Furthermore, the stochastic data augmentation, batch normalization, and weights regularization all contribute to prevent overfitting. The F scores do not increase until around 200 epochs, owing to the fact that the binarization of the probability map was based on a high threshold designed to extract spines that occupy very small regions of the image. Also, it was observed that the onset of increasing F scores occurred later in ESPC networks than in FSC networks.

3.5. Performance on test data

The precision and recall evaluated on the test sets using ESPC-4 are plotted by varying β with $\delta = 16$ pixels, as shown in Fig. 7 (A). Annotations from both readers are used as the reference standard. The F scores on the test sets are 0.822 and 0.847 from the two readers respectively. The values of precision and recall against both readers with fixed $\beta = 200$ and varying δ are shown in Fig. 7 (B). When the tolerance δ is greater than 6 pixels, the precision and recall values are stable. When δ is smaller than 6 pixels, the performance decreases largely with

Table 1
Comparison of Precision, Recall, and F scores in validation data using different architectures before and after the extraction of dendritic shaft.

Architectures	Reader 1						Reader 2					
	Before Shaft Extraction			After Shaft Extraction			Before Shaft Extraction			After Shaft Extraction		
	P	R	F	P	R	F	P	R	F	P	R	F
FSC-8	0.73	0.89	0.80	0.74	0.86	0.79	0.75	0.87	0.80	0.76	0.84	0.80
FSC-4	0.70	0.91	0.79	0.74	0.89	0.80	0.72	0.88	0.79	0.76	0.86	0.81
FSC-2	0.73	0.81	0.77	0.75	0.78	0.76	0.76	0.80	0.78	0.78	0.78	0.78
ESPC-8	0.68	0.87	0.77	0.78	0.86	0.82	0.70	0.84	0.76	0.79	0.83	0.81
ESPC-4	**0.79**	0.88	**0.83**	**0.80**	0.86	**0.83**	**0.83**	0.87	**0.85**	**0.83**	0.84	**0.83**
ESPC-2	0.66	0.90	0.76	0.74	0.89	0.81	0.68	0.88	0.77	0.77	0.87	0.81
GoogleNet FCN-8s	0.72	0.86	0.79	0.76	0.85	0.80	0.75	0.85	0.80	0.79	0.84	0.82
VGG16 FCN-8s	0.77	0.89	0.82	0.78	0.87	0.82	0.78	0.86	0.82	0.79	0.84	0.82
UNet	0.73	**0.94**	0.82	0.74	**0.91**	0.82	0.76	**0.93**	0.84	0.77	**0.90**	0.83

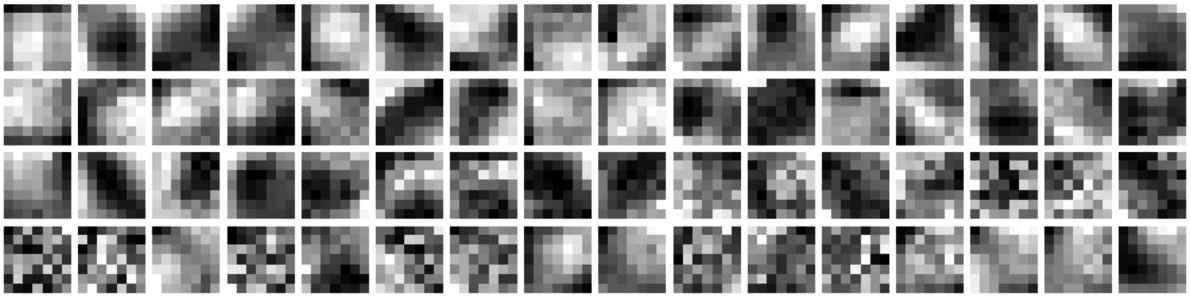

Fig. 4. Filters in the first convolutional layer of ESPC-4. Each filter has size 7 × 7, showing the low-level features learned by the network. The filters are sorted from high variance to low variance.

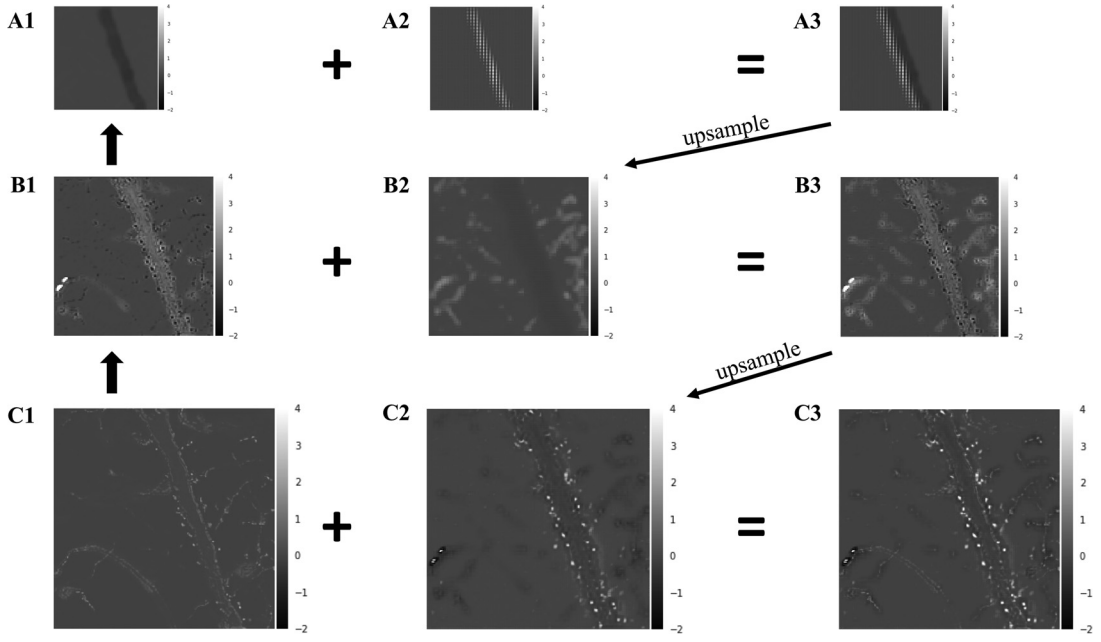

Fig. 5. The second channel of the output of selected layers (or activation maps) in the ESPC-4 model applied on an image from validation data. The x and y dimensions of images in rows A, B, and C, are in accordance with the red, orange and purple layers in Fig. 3, respectively: Row A: 64 × 64, Row B: 128 × 128, and Row C 256 × 256 pixels. Flow from Fig. 3: C1-B1-A1-A2-A3-B2-B3-C2-C3. C1 is from the lowest layer among the nine, showing edges of the all structures in the image. B1 is from the convolution on the orange block before pool3, with lower resolution than C1 and higher contrast between spines and dendritic shaft. A1 has even lower resolution, and the details about spines are lost. Starting from A2, the images are from the up-sampling half of the architecture. A2 exhibit the checkerboard patterns known to exist in such networks. The summation of A1 and A2 leads to A3. B2 is from the convolutional up-sampling from A3. The summation of B1 and B2 leads to B3, and C2 is from the convolutional up-sampling from B3. We can see from C2 that the spines have higher values than the background pixels. Finally, the summation of C1 and C2 leads to C3, during which the local information in C1 is combined with the global information in C2 to generate C3. In the case of ESPC-4, the layer corresponding to C3 gets up-sampled directly to the output of the network. For the purpose of visualization, values in all images are clipped from -2 to 4.

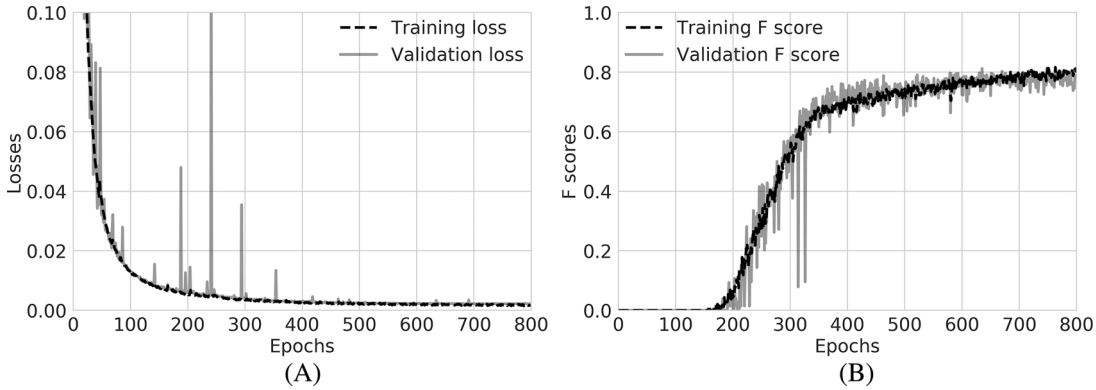

Fig. 6. Training and validation losses (A) and F scores (B) vs. training epochs before shaft extraction evaluated based on Reader 1 using ESPC-4.

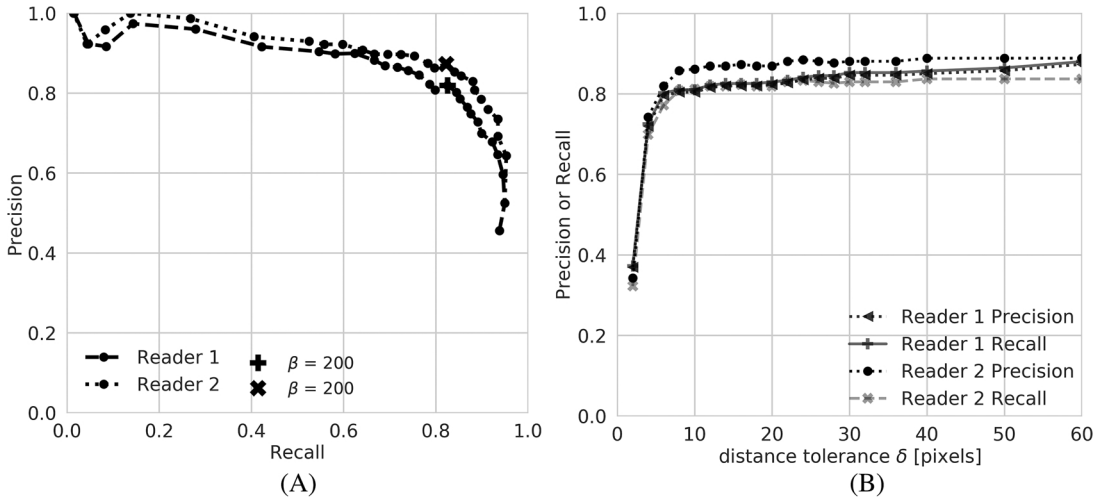

Fig. 7. Performance on test data. Precision-Recall curve with changing parameter β and fixed $\delta = 16$ pixels (A). Precision and recall values with changing parameter δ and fixed $\beta = 200$ (B). Both are plotted using two sets of annotations.

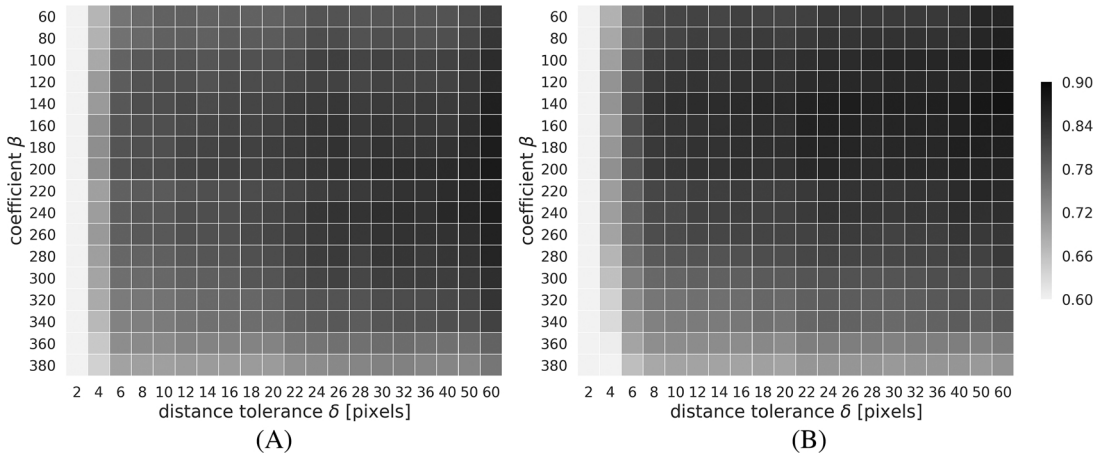

Fig. 8. Heat map of F-scores with varying parameters (β and δ) evaluated based on Reader 1 (A) and Reader 2 (B).

decreasing tolerance. This means that stricter evaluation criteria have an effect only when the distance tolerance is below 6 pixels.

The influence exerted by the variation in β and δ on F scores in test data is illustrated in the heat maps in Fig. 8, for both Reader 1 (A) and Reader 2 (B). The F scores are relatively invariant to changes of β between 160 and 220 and of δ between 12 and 18 pixels (Reader 1: 0.81 ± 0.0077, Reader 2: 0.84 ± 0.0086) for both readers.

The averaged distance between the predicted true spines and annotated ones is 2.81 ± 2.63 pixels (0.082 ± 0.076 microns) based on Reader 1 and 2.87 ± 2.33 pixels (0.084 ± 0.068 microns) based on Reader 2. When evaluating one reader's annotations against another, the F score is 0.925, indicating human performance is still better, but comparable to the automated one.

Three examples of the process are shown in Fig. 9., starting with the input de-convolved MIP images (Fig. 9A), through to the prediction results before and after shaft extraction (Figs. 9D and E). Spines that are too remote from the main shaft are not considered in detection. Despite the fact that the same spine missed in one MIP image may be detected

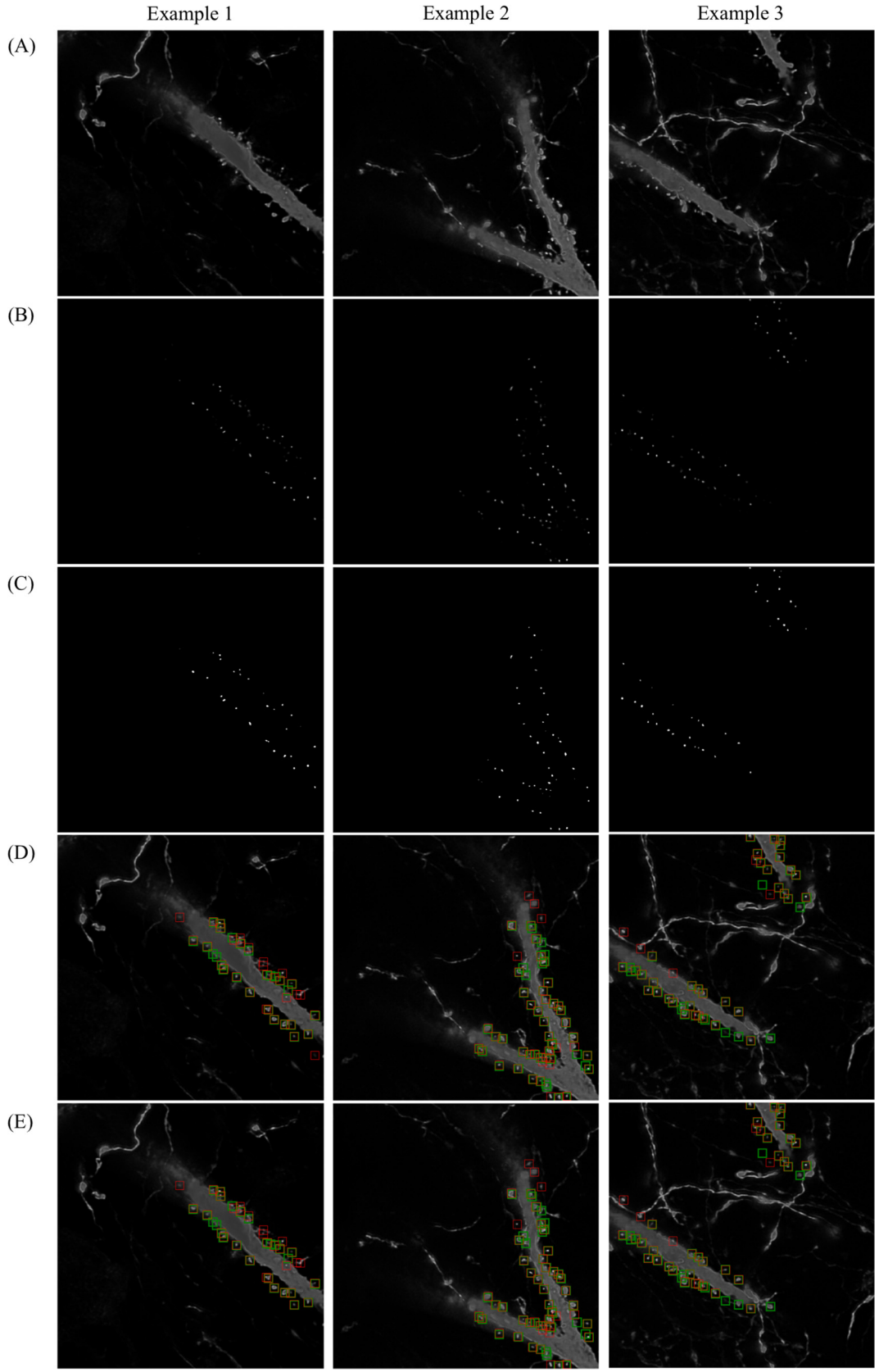

Fig. 9. Prediction results from three example images. (A) De-convolved MIP image, (B) Predicted probability map output from a trained model, (C) Binarized prediction from a trained model, (D) Predicted spines *before* shaft extraction (centers of red boxes) overlaid on a de-convolved MIP image with reader 1's annotations (centers of green boxes), (E) Predicted spines *after* shaft extraction (centers of red boxes) overlaid on a de-convolved MIP image with reader 1's annotations (centers of green boxes). The size of the bounding boxes is fixed at 32 pixels and used only for visualization purposes. True positive predictions (indicated by red boxes that overlap with green boxes) are generally very close to the corresponding seed pixels. The examples shown cover the challenging cases of multiple branches of dendrites, faint spines, and small branches closer to the edges of images, as well as existence of axons and boutons that are potential false positives.

Table 2
Comparison of our method with two existing software packages evaluated by both readers.

	TP		FP		FN		Precision		Recall		F score	
	Reader 1	Reader 2	Reader 1	Reader 2	Reader 1	Reader 2	Reader 1	Reader 2	Reader 1	Reader 2	Reader 1	Reader 2
NeuronStudio	133	142	204	194	125	134	0.396	0.423	0.516	0.515	0.448	0.464
Neurolucida 360	222	212	212	202	46	54	0.500	0.524	0.822	0.804	0.622	0.634
Our method	227	213	47	33	45	49	0.819	0.873	0.826	0.822	0.822	0.847

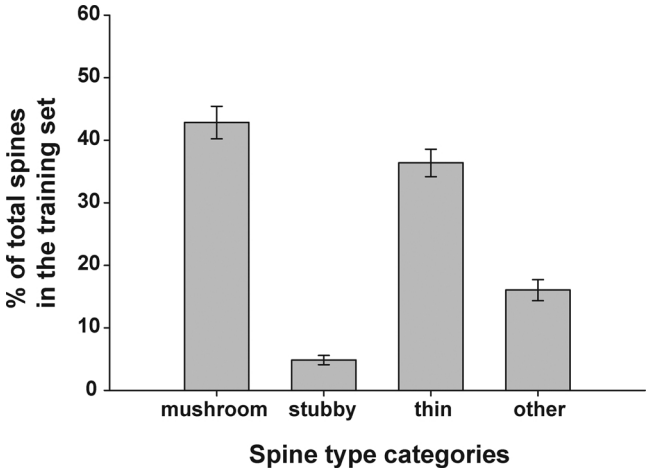

Fig. 10. Distribution of spine types in our dataset.

in another MIP image, they were regarded as different spines during evaluation. Not every spine is at the vicinity of the dendritic shaft in MIP images, indicating the need for three-dimensional analysis to avoid such false negatives.

3.6. Comparison with other dendritic spine detection methods

As described in Section 2.5, dendritic spine detection results using our method are compared against the results obtained from two available software packages, NeuronStudio (NS) and Neurolucida 360 (NL). Software and/or data from other related published methods (Bai et al., 2007; He et al., 2012; Su et al., 2014; Yuan et al., 2009) were not made available to us for comparison. Table 2 summarizes the results, comparing the total number of true positives (TP), false positives (FP), false negatives (FN), precision, recall, and F scores between our pipeline, NS and NL. Using our ESCP-4 pipeline, F score was robustly increased by 83% relative to NS, and 32% when compared to NL. These increases are significant; $p = 0.018$ for both NS and NL vs. our method, using Wilcoxon signed-rank tests based on F scores per image. The main improvement from our pipeline is from about five times fewer false positives relative to both NS and NL, and about twice as many true positives and two-fold fewer false negatives when compared to NeuronStudio. We hypothesize that most of the differences are due to the fact that thin and faint spines are often missed in the software. Even when compared to NL, which has comparable recall values to our method, our method detects 31 more spines from the "thin" category out of the total of 213 true positives based on Reader 1's annotations. According to the distribution of spine categories in our dataset (Fig. 10), thin spines constitute almost 40% of the total number of spines; by extension, undercounting thin spines would contribute to a significant underestimation of total spine numbers, as well as influence conclusions on the involvement of thin spines in learning.

Because of heavy reliance on the shaft extraction, NS and NL struggle to distinguish a dendritic spine from a neuronal structure of similar morphology close to the traced skeleton, resulting in more false positives and lower precision. Furthermore, the shaft extraction in NS and NL introduces error when part of the shaft has low intensity. In this case, manual adjustment on the shaft is needed, and is usually not perfect. In contrast, we have shown that shaft extraction is not indispensable as an additional post-processing step when FCN pipeline is used, thus reducing the amount of manual intervention needed for accuracy, and making the spine detection closer to fully automated. Together, our results suggest that FCNs, and ESPC-4 model applied to the test-set, offer significant improvements in the accuracy of dendritic spine detection when compared to other frequently used methods, an improvement which is, in addition to the automation, known to be achievable with deep learning approaches.

4. Conclusions and future work

We present an automated pipeline for 2D dendritic spine detection using fully convolutional neural networks with different up-sampling schemes, as a pilot approach for future 3D detection efforts. Unlike other approaches that can be limited to datasets or certain types of spines, good predictive models in machine learning are capable of generalizing. In addition, our approach works even on a small set of images, with the flexibility to adapt to the increasing need of processing large datasets of dendritic spine images. Unlike the approaches that design the discriminative features of spines using mathematical models, our deep networks can be trained from end-to-end and learn the features automatically. With trained models, our method requires minimal parameter tuning. Furthermore, instead of heavily relying on finding the exact boundaries of the dendritic shaft as in some previous efforts, we found that dendritic shaft extraction for pruning false positives in the post-processing stage is not critical with FCNs. The performance of our model significantly exceeds that of the semi-automated versions of NeuronStudio and Neurolucida 360.

To our knowledge, our method is the first to apply convolutional neural networks to the task of dendritic spine detection. The deep learning approach is advantageous since it can learn a complex image recognition task, and it performs well. However, there are some limitations of our study, and improvements and extensions could be made to enhance the technique. First, training deep networks can be time-consuming, and requires a large amount of manually labelled images. To ameliorate the latter, we used approximated masks as our ground truth and performed detection using a method designed for pixel-level segmentation tasks where each pixel is a training example. The evaluation of the method is therefore on a per-spine basis instead of pixel-wise. If more annotated data from different imaging modalities were provided, we could train a more powerful model. Second, spines may not retain the same morphology in MIP images as in each slice, let alone in three dimensions. Also, some spines imaged on top of the dendritic shaft were neglected in MIP images. To address this limitation, we are currently working on extending our method to three-dimensional image data.

There are important research applications of our work. Once dendritic spines are detected by our methods, additional image processing can be performed to extract image features for spine analysis of each detected spine, such as segmentation and spine density calculation. These features can then be correlated with biological or clinical aspects of the tissue to glean biological insights into functional changes in dendritic spines that may heretofore been overlooked.

Acknowledgements

This research was supported by a collaborative seed grantfrom the Stanford Bio-X program to D.R. and C.J.S. Spine images were derived from experimental material obtained from M.D. and C.J.S. supported by NIH grant EY02858.

Appendix A. Supplementary data

Supplementary material related to this article can be found, in the online version, at doi:https://doi.org/10.1016/j.jneumeth.2018.08.019.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., 2016. TensorFlow: a system for large-scale machine learning. OSDI 16, 265–283.

Bai, W., Zhou, X., Ji, L., Cheng, J., Wong, S.T., 2007. Automatic dendritic spine analysis in two-photon laser scanning microscopy images. Cytom. Part A 71, 818–826.

Blumer, C., Vivien, C., Genoud, C., Perez-Alvarez, A., Wiegert, J.S., Vetter, T., Oertner, T.G., 2015. Automated analysis of spine dynamics on live CA1 pyramidal cells. Med. Image Anal. 19, 87–97. https://doi.org/10.1016/j.media.2014.09.004.

Christ, P.F., Elshaer, M.E.A., Ettlinger, F., Tatavarty, S., Bickel, M., Bilic, P., Rempfler, M., Armbruster, M., Hofmann, F., D'Anastasi, M., Sommer, W.H., 2016. Automatic liver and lesion segmentation in CT using cascaded fully convolutional neural networks and 3D conditional random fields. International Conference on Medical Image Computing and Computer-Assisted Intervention 415–423.

Day, M., Wang, Z., Ding, J., An, X., Ingham, C.A., Shering, A.F., Wokosin, D., Ilijic, E., Sun, Z., Sampson, A.R., Mugnaini, E., 2006. Selective elimination of glutamatergic synapses on striatopallidal neurons in Parkinson disease models. Nat. Neurosci. 9 (2), 251–259.

Dickstein, D.L., Dickstein, D.R., Janssen, W.G.M., Hof, P.R., Glaser, J.R., Rodriguez, A., O'Connor, N., Angstman, P., Tappan, S.J., 2016. Automatic dendritic spine quantification from confocal data with Neurolucida 360. Curr. Protoc. Neurosci. 77, 1.27.1–1.27.21.

Djurisic, M., Vidal, G.S., Mann, M., Aharon, A., Kim, T., Santos, A.F., Zuo, Y., Hübener, M., Shatz, C.J., 2013. PirB regulates a structural substrate for cortical plasticity. Proc. Natl. Acad. Sci. 110 (51), 20771–20776.

Fan, J., Zhou, X., Dy, J.G., Zhang, Y., Wong, S.T., 2009. An automated pipeline for dendrite spine detection and tracking of 3d optical microscopy neuron images of in vivo mouse models. Neuroinformatics. 7, 113–130.

Feng, G., Mellor, R.H., Bernstein, M., Keller-Peck, C., Nguyen, Q.T., Wallace, M., Nerbonne, J.M., Lichtman, J.W., Sanes, J.R., 2000. Imaging neuronal subsets in transgenic mice expressing multiple spectral variants of GFP. Neuron 28 (1), 41–51.

Fu, M., Yu, X., Lu, J., Zuo, Y., 2012. Repetitive motor learning induces coordinated formation of clustered dendritic spines in vivo. Nature 483 (7387), 92–95.

Graveland, G.A., Williams, R.S., DiFiglia, M., 1985. Evidence for degenerative and regenerative changes in neostriatal spiny neurons in Huntington's disease. Science 227 (4688), 770–773.

Grutzendler, J., Helmin, K., Tsai, J., Gan, W.B., 2007. Various dendritic abnormalities are associated with fibrillar amyloid deposits in Alzheimer's disease. Ann. N. Y. Acad. Sci. 1097 (1), 30–39.

He, T., Xue, Z., Wong, S.T., 2012. A novel approach for three-dimensional dendrite spine segmentation and classification. SPIE Med. Imaging. 8314https://doi.org/10.1117/12.911693. 831437-831437.

He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: surpassing human-level performance on imagenet classification. Proceedings of the IEEE International Conference on Computer Vision. pp. 1026–1034.

Ioffe, S., Szegedy, C., 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. International Conference on Machine Learning. pp. 448–456.

Janoos, F., Mosaliganti, K., Xu, X., Machiraju, R., Huang, K., Wong, S.T., 2009. Robust 3d reconstruction and identification of dendritic spines from optical microscopy imaging. Med. Image Anal. 13, 167–179.

Johnson, J., Alahi, A., Fei-Fei, L., 2016. Perceptual losses for real-time style transfer and super-resolution. European Conference on Computer Vision 694–711.

Kamnitsas, K., Ledig, C., Newcombe, V.F.J., Simpson, J.P., Kane, A.D., Menon, D.K., Rueckert, D., Glocker, B., 2017. Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. Med. Image Anal. 36, 61–78. https://doi.org/10.1016/j.media.2016.10.004.

Kingma, D., Ba, J., 2014. Adam: a method for stochastic optimization. arXiv preprint arXiv:1412. 6980.

LeCun, Y., Bottou, L., Orr, G.B., Müller, K.R., 1998. Efficient backprop. In Neural Networks: Tricks of the Trade. Springer, Berlin Heidelberg, pp. 9–50.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3431–3440.

Majewska, A., Sur, M., 2003. Motility of dendritic spines in visual cortex in vivo: changes during the critical period and effects of visual deprivation. Proceedings of the National Academy of Sciences 100 (26), 16024–16029.

Mukai, H., Hatanaka, Y., Mitsuhashi, K., Hojo, Y., Komatsuzaki, Y., Sato, R., Murakami, G., Kimoto, T., Kawato, S., 2011. Automated analysis of spines from confocal laser microscopy images: Application to the dis- crimination of androgen and estrogen effects on spinogenesis. Cereb. Cortex 21, 2704–2711.

Rodriguez, A., Ehlenberger, D.B., Dickstein, D.L., Hof, P.R., Wearne, S.L., 2008. Automated three-dimensional detection and shape classification of dendritic spines from fluorescence microscopy images. PLoS One 3, e1997. https://doi.org/10.1371/journal.pone.0001997.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: convolutional networks for biomedical image segmentation. International Conference on Medical Image Computing and Computer-Assisted Intervention 234–241.

Shankar, G.M., Bloodgood, B.L., Townsend, M., Walsh, D.M., Selkoe, D.J., Sabatini, B.L., 2007. Natural oligomers of the Alzheimer amyloid-ß protein induce reversible synapse loss by modulating an NMDA-type glutamate receptor-dependent signaling pathway. J. Neurosci. 27 (11), 2866–2875.

Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z., 2016. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1874–1883.

Su, R., Sun, C., Zhang, C., Pham, T.D., 2014. A novel method for dendritic spines detection based on directional morphological filter and shortest path. Comput. Med. Imaging Graph. 38, 793–802.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–9.

Yaeger, L.S., Lyon, R.F., Webb, B.J., 1997. Effective training of a neural network character classifier for word recognition. Adv. Neural Inf. Process. Syst. 807–816.

Yuan, X., Trachtenberg, J.T., Potter, S.M., Roysam, B., 2009. MDL constrained 3-d grayscale skeletonization algorithm for automated extraction of dendrites and spines from fluorescence confocal images. Neuroinformatics 7, 213–232. https://doi.org/10.1007/s12021-009-9057-y.

Zhang, Y., Zhou, X., Witt, R.M., Sabatini, B.L., Adjeroh, D., Wong, S.T., 2007. Dendritic spine detection using curvilinear structure detector and LDA classifier. Neuroimage 36, 346–360.

Zhang, Y., Chen, K., Baron, M., Teylan, M.A., Kim, Y., Song, Z., Greengard, P., Wong, S.T., 2010. A neurocomputational method for fully automated 3d dendritic spine detection and segmentation of medium-sized spiny neurons. Neuroimage 50, 1472–1484.

Zuo, Y., Lin, A., Chang, P., Gan, W.B., 2005a. Development of long-term dendritic spine stability in diverse regions of cerebral cortex. Neuron 46 (2), 181–189.

Zuo, Y., Yang, G., Kwon, E., Gan, W.B., 2005b. Long-term sensory deprivation prevents dendritic spine loss in primary somatosensory cortex. Nature 436 (7048), 261–265.