Robust Neuron Counting Based on Fusion of Shape Map and Multi-cue Learning

Alexander Ekstrom[1], Randall W. Suvanto[1], Tao Yang[2], Bing Ye[2], and Jie Zhou[1(✉)]

[1] Department of Computer Science, Northern Illinois University, DeKalb, IL 60115, USA
{zl664374, zl779093}@students.niu.edu, jzhou@niu.edu
[2] Life Sciences Institute and Department of Cell and Developmental Biology, University of Michigan, Ann Arbor, MI 48109, USA
{taoyang, bingye}@umich.edu

Abstract. Automatic counting of neurons in fluorescently stained microscopic images is increasingly important for brain research when big imagery data sets are becoming a norm and will be more so in the future. In this paper, we present an automatic learning-based method for effective detection and counting of neurons with stained nuclei. A shape map that reflects the boosted edge and shape information is generated and a learning problem is formulated to detect the centers of stained nuclei. The method combines multiple cues of edge gradient, shape, and texture during shape map generation, feature extraction and final count determination. The proposed algorithm consistently delivers robust count ratios and precision rates on neurons in mouse and rat brain images that are shown to be better than alternative unsupervised and supervised counting methods.

Keywords: Neuron counting · Machine learning · Shape map · Microscopic neuronal image · Nuclei staining

1 Introduction

Along with the recent advancement of imaging and florescent staining, automatic counting of neurons using microscopic images is seeing increasingly important applications in neuroscience including developmental neuroscience, study of the neural functions, as well as the effects of neural diseases and their cures [1–3]. When the volume and the scale of the neural imagery data grow, as the inevitable trend at the big data era, automatic counting of cells and neurons is expected to be more essential in scientific discovery. To promote and evaluate the computational methods for automatic neuron counting, several competitions have been held, with a recent example being the Nucleus Counting Challenge at the 2015 Bioimage Informatics Conference.

However, obtaining a robust neuron count remains a very challenging task for several reasons. First of all, the staining of the neurons can be fuzzy due to the imaging technique. Second, intensity and patterns vary greatly among different types of staining labels. Even for the same type of nuclear label, there often exist intensity variations

G.A. Ascoli et al. (Eds.): BIH 2016, LNAI 9919, pp. 3–13, 2016.
DOI: 10.1007/978-3-319-47103-7_1

among different nuclei or within the same nucleus, which make it difficult to tell them apart from background and artifacts. In addition to the complexity and variability of morphology and intensity of the stained objects, cells or nuclei are often clustered in the image, which further increases the level of difficulty for automatic counting. As a result, current automatic software for cell counting falls short on providing the robustness needed in neuroscience research. Figure 1 shows the images of several popular staining of nuclei. We can see non-uniform and fuzzy objects from multiple staining techniques.

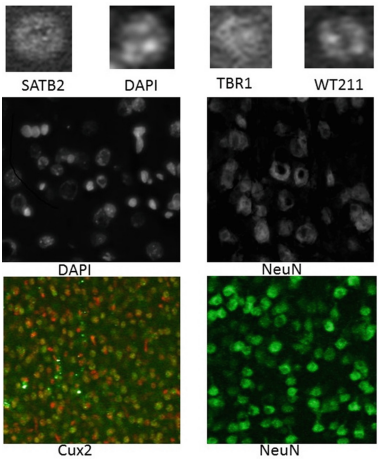

Fig. 1. Nuclei staining. Images are from BII 2015 nuclei counting challenge or Bing Ye lab. Some are ubiquitous stains of all nuclear DNA (e.g. DAPI) and some are only expressed in neurons (e.g. NeuN).

A majority of current automatic counting methods for neurons – and for cells in general – are based on the principle of image segmentation. The stained components of different neurons (often nuclei) are considered as the individual objects to be segmented from the image background and separated apart from each other. Counting is then performed after the segmentation. The specific method varies from local or global thresholding to region-based watershed to dynamic models [1, 2]. Counting methods relying on image processing/segmentation are unsupervised traditionally. A relatively recent category of methods, on the other hand, makes use of supervised machine learning. A cell counting method using machine learning starts by labeling some training samples. The counting is then performed using pattern recognition [3, 4].

While both methods have their uses, they also have shortcomings. The unsupervised category suffers from many traditional obstacles of image segmentation, which has been a knowingly difficult problem in image processing, especially with fuzzy and/or clustered objects in images. It also requires more parameter tuning to overcome over-segmentation or under-segmentation. In contrast, the learning-based approach for cell counting builds the model using known examples. It potentially requires fewer

critical parameters. It also has the advantage of a more flexible problem formulation if full segmentation is not needed for counting. Yet pattern recognition algorithms are trained based on the features extracted from the training samples. The relevance of the features directly impacts the result of detection. For example, features extracted from the intensities of regions of interest may not make full use of the object properties such as shape and neighborhood gradient.

In this paper, we report an approach that aims to combine these two methods to achieve improved robustness for automatic counting of neurons using microscopic images with florescent staining. The core of the approach is a learning-based detection for the centers of stained nuclei. As a hybrid solution that combines traditional segmentation and machine learning, the proposed method is able to integrate multiple cues including neuron edges, shapes, intensity distribution and texture, and achieve a robust count for images with intensity variety and clustered objects. We will explain our method in Sect. 2, followed by experimental results and discussions.

2 Method

2.1 Rationale

For microscopic images with staining of neuron nuclei such as STAB2 or DAPI as shown in Fig. 1, the neurons are counted using the stained nuclei.

Traditional counting methods using segmentation approaches such as thresholding or watershed are insufficient for the problem of nuclei counting. Figure 2 shows some example results. We can see that simple thresholding such as Fig. 2B is apparently insufficient. Figure 2C and D show that, on one hand, bright noise is counted by watershed leading to false positives. On the other hand, clustered cells especially those close together will be counted as one cell. This leads to false negatives.

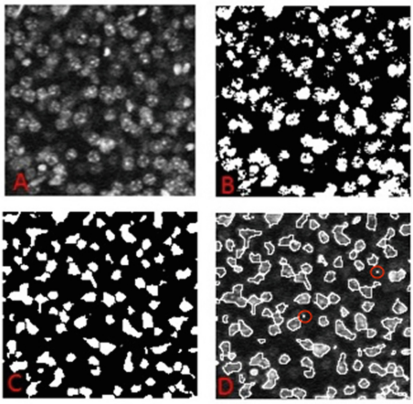

Fig. 2. Effects of common segmentation methods. (A) Raw image (B) ImageJ thresholding (C) and (D) Watershed and the resulting counting. False positives are circled in D for annotation.

Learning-based models have also been experimented. Trainable segmentations that classify every pixel to detect the nuclei have similar issues as other traditional segmentation-based methods for counting. An alternative formulation is to detect the center of stained nuclei, instead of all pixels of the nuclei. It, however, encounters issues when a nuclei stain such as DAPI leads to multiple bright spots in the same cell nucleus. Other stains tend to have the similar problem of uneven staining. This issue tends to lead towards over counting when a cell is counted multiple times as each bright spot is detected as a separate center. However, if the features exacted from the region of interest surrounding the center of nucleus reacts better to shape data, then we hypothesize that the combination of the two methods may lead to better results.

To this end, we decided that the unsupervised stage can be more aggressive to segment the objects and effectively reduce clustering when extracting the object shapes. As long as the machine learning phase can smartly reject the over-segmented parts, it will not lead to over count. Aggressive segmentation also reduces the counting problem to only be a matter of reducing the objects, rather than both reducing and attempting to split. We thus design several steps to achieve this in the proposed algorithm. We incorporate aggressive background subtraction to enhance contrast before edges are extracted and watershed is applied. A distance transform map replaces the original intensity information with distances from edges, which are essentially shape information and are named *shape-map* in the algorithm.

With the motivation to boost a machine learning-based counting method using multiple cues from the image, we propose to apply learning on the *shape-map* image. Compared with the original microscopic image, the shape-map image will be edge-boosted, pre-segmented, and distance-transformed with shape and boundary information better reflected. It will serve as the base for further feature extraction and learning. In addition, multiple cues of nuclei properties such as texture and geometry will be included in the features when a classifier is applied for the detection of nuclei centers. As a result, the proposed method, surrounding a machine learning backbone, provides a fusion of intensity, edge and shape information in the detection and counting of neurons. In addition, to ensure that during the learning step, only the relevant shape information is extracted from the region surrounding the center of nuclei, we also incorporated a flood-mask step to exclude nearby objects.

Summarizing the rationale above, we have the resulting proposed counting algorithm described in Sect. 2.2.

2.2 Algorithm Flow

The proposed method contains three phases: Generate the shape-map image, identify the centers of nuclei using machine learning based detection, and come up with a final neuron count after post processing.

Shape-Map Image Generation. The emphasis of the first stage is to boost the edge and shape information of the original image, and provide a base for the machine learning step.

First, if the image contains clustered objects or is very noisy, an initial background subtraction is performed using a rolling ball algorithm, which corrects or uneven illuminated background, since a cleared background improves the contrast of the edges. A relatively small radius for the rolling ball is used to make this fairly aggressive. The second step is finding edges using Sobel-Feldman filtering [5]. As stated above, this is done to convert the original image into a set of shape data. Several actions taken from this point will be altering and enhancing the shape information. Specifically, a histogram equalization is done followed by a despeckling to remove noise. The edge image is then binarized using local adaptive thresholding, followed by watershed to split the nearby objects. Two final steps are applying the distance transform and another histogram equalization which boosts the distance map image. The whole process results in the shape-map image.

Note that although thresholding and watershed steps are applied in the process of generating the shape-map, the purpose here is to create the image that highlights the edge and shape information. The shape map contains objects that may or may not be the neurons to be counted. However, this carefully designed process converted the original raw image to an image where shapes are emphasized. As the result, the following learning will be less sensitive to uneven intensities within a nucleus, as well as staining variation among different neurons. Figure 3 shows two examples of generated shape maps.

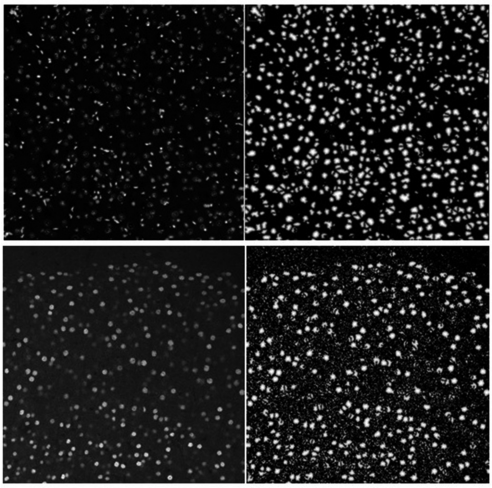

Fig. 3. The shape-map images. Left column: raw images. Right column: shape-maps. Top row: DAPI stained rat brain neurons. Bottom row: STAB2 stained mouse brain neurons

Learning the Centers of Neuron Nuclei. The shape-map images, as shown in Fig. 3, are transformed images that highlight the edge and shape information. A machine learning engine is then applied to the shape-map image for learning.

The pattern recognition formulation at this stage has several possibilities. One is to classify each pixel to see if it belongs to *any* part of nuclei as often employed in trainable segmentation. However, in our context, counting instead of segmentation is the main goal, and shape-map highlights the centers of shapes, so we formulate the problem as the identification of the *center* of the nuclei. It is a binary problem to tell if a pixel is a center of cell nuclei or not.

As needed by supervised learning, the first task is to label the positive and negative samples. Objects in the shape map image may correspond to actual cells or noise arising from edges of background artifacts. The task of labeling training samples is to tell apart two types of objects. We developed an ImageJ plugin for this purpose as shown in Fig. 4. For an improved usability, the tool provides an automatic mapping of the selected regions in the shape-map back to the raw image, so that the user can confidently decide positive or negative objects on the shape map. For cells that were split into many pieces during the generation of shape-map, the tool marks the largest subsection positive, and the rest as negative. This is to facilitate the classifier to pick out only one representative portion of each cell from the shape-map in the detection stage.

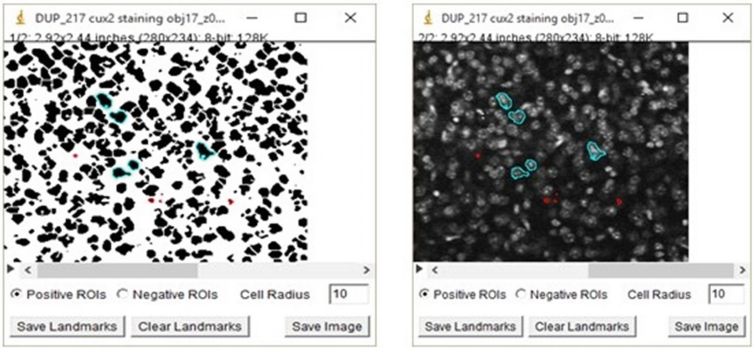

Fig. 4. Labeling positive and negative training samples on the shape-map image. The cyan-circled shapes are the positive objects correspond to cells in the raw image. The red ones are the negative ROIs. The user selects objects on the shape-map image (left panel), and corresponding regions on the raw image are automatically marked (right panel) to facilitate the labeling

About ten positive samples and ten negative samples are chosen for training. Features are extracted from the surrounding regions of each center pixel of the sample ROI. The size of the surrounding regions is set based on the expected nuclei size. Assume the radius of the expected nuclei is r, then the neighborhood size is (2r + 1) * (2r + 1). This region centered around the pixel-in-question is used for feature extraction, with a modification: since the object shape is what is being dealt with now, it is important not to include nearby objects in the case of clusters. A flood masking is applied starting from the center and stopping at the object boundary, so that only data from one object in the region is used for feature extraction. This prevents nearby objects from confounding the details of the object of question. Multiple features are then extracted from the modified surrounding region. The following features are used:

- Wavelet-based texture features:

$$f(k,n) = \iint \emptyset_{k,n}(i,j)I(i,j)didj$$

where ϕ is the two dimensional discrete wavelet Haar base function [6]. k is set to 0,1 for two level transform. n is the index at the specific level, n = 1 … $|i| * |j|$, where $|i|$ and $|j|$ are the width and height of the sub-image analyzed at the level. I(i, j) is the intensity at the location of the subimage for the specific level. The total number of texture features equals to the size of the surrounding region.

- The Gaussian correlation coefficient with a Gaussian template

$$f = \iint I(x,y) * \theta(x,y)dxdy$$

where Θ (x, y) is intensity at location (x, y) of the 2D Gaussian approximation centered at the pixel of question, with σ set to the expected radius of the nuclei r.

- Four image statistics are used. Geometry symmetry at x, y directions, as well as the mean and standard deviation of the region intensities.

The total number of features depends on the nuclei radius parameter r which decides the size of ROI centered by the pixel of question. For a radius of 3 and a surrounding region of 7 * 7, the number of features is 54. Learning is then performed by training a support vector machine with a linear kernel [7] to maintain speed efficiency without sacrificing reliability.

To detect of the centers of nuclei in an image, foreground pixels of the shape-map are fed to the learned classifier. If the prediction is positive for the pixel, it is a candidate to be a nuclei center. The pixels of positive candidates are then mean-shifted to the closest center of mass, and merged with others if multiple predictions are shifted to the same center of mass. The resulting number yields the tentative count of cells.

A post-processing is then performed to finalize the count. The post-processing stage links the results obtained from the shape-map image back to the original florescent stained image to perform double-checking. Mean shifting of the detected centers is re-attempted in the original image based on the florescent intensity to offset the effects of aggressive splitting. In addition, intensities surrounding the detected center in the original image are examined to avoid dim spots from being counted in order to alleviate false positives.

3 Results and Discussions

The following measures are used to quantitatively assess the proposed counting algorithm: Count Ratio, Precision and Recall. Manually counted results are used as a golden standard in calculation.

Count ratio is defined as the ratio of automatically counted neurons to the golden standard count. We define a detected cell as precise if the computationally detected nuclei center is within the boundary of a cell marked manually. The precision is then

calculated as the ratio of precise cells to all detected cells. The recall rate is calculated as the ratio of true positive cells to total golden standard cells.

Experimental Results. We report our results on two types of stained neuronal images as shown in Fig. 3. One is STAB2 stained mouse brain neurons, the other is the DAPI stained rat brain neurons. Table 1 lists the count ratios. Table 2 lists the precisions and recalls obtained on these images. From Tables 1 and 2, we can see that the proposed neuron counting algorithm delivers a robust count ratio of 95 % or above on the STAB stained mouse brain images, and an average 90 % count ratio on the NeuN stained rat brain image. It also consistently delivers precision mostly around or above 95 %.

Table 1. Count ratios. Different rolling ball radii for background subtraction are tested. Image 1–6 are STAB2 stained. Image 7 is a DAPI-stained image from BII 2015 nuclei counting challenge. Radius for image 1–6 is set to 3 pixels. For image 7 the radius is set to 7 pixels.

Image	1		2		3		4		5		6		7	
Standard	312		347		324		325		319		305		801	
Rolling ball	Count	Ratio	Count	Ratio	Count	Ratio	Count	Ratio	Count	Ratio	Count	Ratio	Count	Ratio
10	301	0.96	337	0.97	317	0.98	297	0.91	284	0.89	284	0.93	756	0.94
20	326	1.04	332	0.96	300	0.93	299	0.92	306	0.96	293	0.96	753	0.94
30	319	1.02	341	0.98	325	1.00	311	0.96	311	0.97	319	1.05	672	0.84
40	313	1.00	336	0.97	325	1.00	323	0.99	312	0.98	261	0.86	704	0.88
Average	315	1.01	337	0.97	317	0.98	308	0.95	303	0.95	289	0.95	721	0.90

Table 2. The precisions and recalls of the STAB2-stained images using different expected nuclei radius.

Image	Rolling ball size	Radius	3	5	7	10	Avg.
1	10	Precision	0.95	0.96	0.96	0.98	0.96
		Recall	0.91	0.90	0.88	0.90	0.90
2	20	Precision	0.93	0.95	0.95	0.95	0.95
		Recall	0.89	0.87	0.84	0.82	0.86
3	30	Precision	0.92	0.95	0.95	0.98	0.95
		Recall	0.92	0.89	0.87	0.85	0.88
4	40	Precision	0.94	0.96	0.96	0.93	0.95
		Recall	0.94	0.87	0.84	0.78	0.86
5	20	Precision	0.90	0.94	0.95	0.96	0.94
		Recall	0.86	0.84	0.81	0.79	0.83
6	30	Precision	0.91	0.93	0.95	0.97	0.94
		Recall	0.95	0.86	0.86	0.83	0.88
Avg.		Precision	0.93	0.95	0.95	0.96	0.95
		Recall	0.90	0.87	0.85	0.82	0.86

The shape-image generation and the labeling tool were implemented in Java as an ImageJ plugin. The learning and counting were implemented in C++ as a Vaa3D plugin [8]. On a typical laptop (2.3 GHz i5 processor with 4 GB of RAM), the algorithm counts 700 cells in about 3 s.

Algorithm Comparison. Figure 5 shows the detected cells on a STAB-stained image using the proposed method, as well as two alternatives. One alternative method is the unsupervised watershed in ImageJ and the other is the same machine learning based formulation but applied directly to the raw image without shape map.

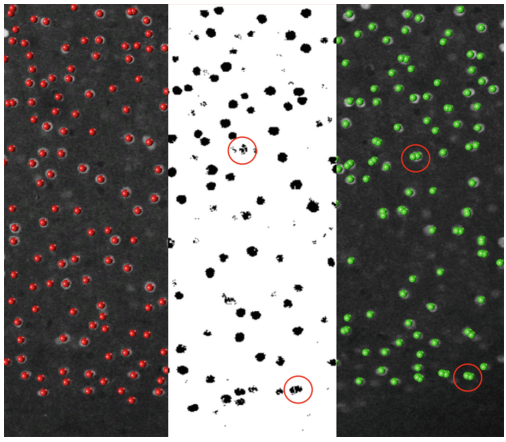

Fig. 5. Counting results and algorithm comparison on various images. The left panel: the proposed method. The middle panel: watershed-based counting. The right panel: machine learning based counting without shape map. The red circles indicate examples of over count.

From Fig. 5, we can see that the two alternative methods suffer from several issues. The watershed algorithm splits several cells into multiple components, resulting in over-count. For example, on test image 1, when the expected radius is set to 5, 18 % more cells were counted by the watershed approach. Meanwhile it also misses some dim objects. Figure 5 shows that the machine learning algorithm by itself tends to overcount as well, due to detecting multiple positive centers when the staining is uneven. The precisions of the alternative algorithms are also lower. We calculated the learning-based algorithm's precision for radius of 3, 5, 7 and 10 on the same testing images as in Table 2. The average of the precision range from 85 % to 89 % for the STAB-stained images, which are significantly lower than the proposed method that achieved an average of 95 % precision on the same images. The similar phenomenon was observed on the NeuN stained rat brain image. Detailed statistics of alternative methods are omitted due to space limit.

Several parameters are used in the algorithm. Most importantly the rolling ball size for clearing the background, and the expected nuclei radius used in feature extraction, mean-shifting, and the calculation of precision. As shown by Table 1, the rolling ball size seems to be a relatively insensitive although some variations are observed. The nuclei radius parameter impacts the results in multiple ways and should be determined carefully based on the expected size of the stained nuclei: With a small expected radius, more cells could be counted, partially related to a smaller region during mean-shifting.

Meanwhile, the measure of precision is stricter with a smaller nuclei boundary. As a result, a smaller radius sees relatively higher recall but a lower precision rate. On the other hand, if the expected radius is set to be larger, the cell counts are lower which reduces the recall as the result.

In addition to the need of choosing the proper nuclei radius, the proposed method also expects that the objects are of somewhat uniform size. While the stained neuron nuclei satisfy this requirement in general, the use of the algorithm to count other types of objects can be less effective if their sizes are dramatically different. The proposed algorithm can be extended to 3D. Some cells are uncounted in the testing images because the confocal imaging focused on a different z-layer with respect to their optimal z-position. It caused some ambiguity during manual labeling. We will experiment using the extended algorithm to count the neurons in the three dimensional image, which is expected to alleviate this problem and further improve the recall rates. However, implementing the same image processing techniques in a three dimensional image will prove to be challenging due to the fact that many of the image processing algorithms used do not support volumetric data.

4 Conclusions

In this paper, we present an automatic neuron counting algorithm that reliably detects and counts stained neuron nuclei in microscopic images. In addition to providing the advantage of a learning-based approach, the hybrid method also makes use of edge gradient, shape and texture information during decision making, which resulted in improved count ratios and precision rates for neuron counting. The proposed method is expected to provide useful assistance in current and future neuroscience research.

Acknowledgements. We thank Dr. Dragan Maric for providing the image for Bioimage Informatics Conference 2015 Nucleus Counting Challenge. The work was partially supported by NIH NIMH R15 MH099569 (Zhou) and R21 NS094091 from NIH and a Seed Grant from the Brain Research Foundation (Ye).

References

1. Lin, G., Adiga, U., Olson, K., Guzowski, J.F., Barnes, C.A., Roysam, B.: A hybrid 3D watershed algorithm incorporating gradient cues and object models for automatic segmentation of nuclei in confocal image stacks. Cytom. A **56**, 23–36 (2003)
2. Oberlaendera, M., Dercksenb, V.J., Eggera, R., Genselb, M., Sakmanna, B., Hegeb, H.-C.: Automated three-dimensional detection and counting of neuron somata. J. Neurosci. Methods **180**, 147–160 (2009)
3. Zhou, J., Peng, H.: Counting cells in 3D confocal images based on discriminative models. In: ACM Conference on Bioinformatics, Computational Biology and Biomedicine (ACM BCB) (2011)
4. Sanders, J., Singh, A., Sterne, G., Ye, B., Zhou, J.: Learning-guided automatic three dimensional synapse quantification for drosophila neurons. BMC Bioinform. **16**, 1–13 (2015)

5. Sobel, I., Feldman, G.: A 3×3 isotropic gradient operator for image processing. In: The Stanford Artificial Intelligence Project (SAIL) (1968)
6. Mallat, S.: A Wavelet Tour of Signal Processing. Academic, San Diego (1999)
7. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**, 1–27 (2011)
8. Peng, H., Ruan, Z., Long, F., Simpson, J.H., Myers, E.W.: V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets. Nat. Biotechnol. **28**, 348–353 (2010)