DEEP-LEARNING-ASSISTED VISUALIZATION FOR LIVE-CELL IMAGES

Hsueh-Chien Cheng[1] *Antonio Cardone*[1] *Eric Krokos*[1] *Bogdan Stoica*[2]
Alan Faden[2] *Amitabh Varshney*[1]
[1]Department of Computer Science and UMIACS, University of Maryland
College Park, Maryland, USA
[2] STAR Center, University of Maryland School of Medicine
Baltimore, Maryland, USA

ABSTRACT

Analyzing live-cell images is particularly challenging because cells simultaneously move and undergo systematic changes. Visually inspecting live-cell images therefore involves simultaneously tracking individual cells and detecting relevant spatio-temporal changes. The high cognitive burden of such a complex task makes this kind of analysis inefficient and error prone. In this paper, we describe a deep-learning-assisted visualization based on automatically derived high-level features to identify target cell changes in live-cell images. Applying a novel user-mediated color assignment scheme that maps abstract features into corresponding colors, we create color-based visual annotations that facilitate visual reasoning and analysis of complex time-varying live-cell image datasets.

Index Terms— Visualization, deep learning, live-cell images

1. INTRODUCTION

Many studies of complex biological systems rely on live-cell imagery to measure, qualitatively and quantitatively, temporal changes in cell structure and behavior. These biologically-relevant changes, such as protein concentration, cell morphology, and geometry, are closely related to cell proliferation and cell life cycle. Several studies have shown that traumatic brain injury (TBI) causes progressive neurodegenerative changes and consequent dysfunction. The identification of highly-expressed proteins related to TBI has enabled the definition of possible therapeutic strategies to reduce TBI-induced neuronal death [1]. In this work, we focus on identifying neuronal death in live-cell images.

Besides the challenges in recognizing complex objects in microscopy images, analyzing live-cell images is particularly difficult because the cells can move in all directions, including moving vertically across different focal planes, as the images are acquired. Furthermore, slight changes in the focus of a microscope over time, a phenomenon referred to as focus drift, may degrade the image quality significantly. A common practice to accommodate for cell movements and focus drift is taking multiple images at various focal planes (or depths) at each time point; this practice ensures all cells are in-focus in at least some of the images. For a cell at a given time point, a small number of in-focus images may contribute much more useful information than the other (out-of-focus) images in determining the state of the cell.

In the past few years, convolutional neural networks (CNNs) have been successful in many biomedical applications such as microscopy image segmentation [2] and subcellular feature detection [3]. Most modern network architectures resemble that of AlexNet [4], which interleaves convolution layers and pooling layers to reduced spatial resolutions progressively. This arrangement allows the network to learn higher-level features in deeper (later) layers by combining lower-level features in shallower (earlier) layers.

In this paper, we use 3D CNNs to determine the cell state independently at each time point (Section 2.2). Targeting exploratory data analysis, we build an interactive visualization tool that annotates cells by a user-defined transfer function, which maps abstract features representing cell characteristics to semantically-meaningful colors (Section 2.3). This color assignment scheme allows users to create informative visualizations despite their limited understanding of the abstract features, such as the deep features derived automatically by CNNs and other sophisticated hand-crafted features. The tool also creates a visual summary to identify when and where cells change their states (Section 2.4). Although many studies have exploited the power of CNNs for computer vision applications, there have been hardly any studies that explore the potential of CNNs for visualization. In this work, we use these deep features as a set of robust features that allow our color assignment scheme to create annotations depicting high-level concepts, which are not easily defined either manually or by conventional low-level features [5].

2. DEEP-LEARNING-ASSISTED VISUALIZATION

In contrast to the work that applies 3D convolutions to address the temporal dimension of videos [6, 7], here we use 3D convolutions to combine features along the dimension formed by the depths of various imaging focal planes. The CNN there-

fore must learn to differentiate in-focus images and out-of-focus images and selectively combine information when predicting cell states. Ideally, we would want to use the (deep) features derived from the CNN to identify potential intermediate states between the transition of cell states (from alive to dead). These intermediate states can then be used to create a graph-based visualization summarizing cell state transitions [8]. As a first step, we present in this paper a color assignment scheme based on deep features, and then use the assigned colors in our visualization.

2.1. Dataset

We use a time-varying dataset that contains 86 microscopy images of adult mice neurons taken every 15 minutes. The live cells in the first two time points are manually labeled. Each image has two channels: the ordinary phase contrast channel and the fluorescence channel, which measures the mitochondrial potential. For each time point, ten images of resolution 1024×1024 are captured at different focal planes. The data at a time point t can be regarded as a stack of ten images of different depths. In summary, the dataset is a 5D dataset of dimensions $(t, x, y, depth, channel) = (86, 1024, 1024, 10, 2)$.

2.2. Feature extraction

Besides the typical x and y dimensions, we use depth as the third dimension for the 3D convolution [6, 7] in our CNN. The additional depth dimension allows the kernels to combine features derived from images taken at different focal planes. In the following, we describe the CNN we used for predicting the cell state of the center pixel that resides in an input patch of dimensions $(x, y, depth, channel) = (65, 65, 10, 2)$ sampled from the image stack at a given time point.

The CNN is composed of three groups of layers, each containing two $3 \times 3 \times 3$ convolutional layers with 64 kernels followed by a $2 \times 2 \times 2$ max-pooling layer that reduces the spatial resolution in all three dimensions (including depth) by half. We apply batch normalization [9] to all convolutional layers and Parametric Rectified Linear Unit [10] as activation function. The last pooling layer (of the third group) is followed by the first fully-connected layer with 200 neurons, which are then connected to the second fully-connected layer that outputs the prediction probability of cell states. In this work, we solve a binary classification problem (i.e. live cell versus dead cells and others), and therefore the number of outputs of the second fully-connected layer equals two.

A total of 400K samples, divided equally between the two classes (i.e. live cells and others), are used to train the CNN using a stochastic gradient descent solver with momentum and decay equal to 0.9 and 0.0005, respectively. The training procedure ends after 240K iterations using a batch size of 50. We have implemented our CNN using the Caffe framework [11]. After training the CNN, our color assignment scheme uses the 200 outputs extracted from the first fully-connected layer as the pixel representation.

2.3. Deep-feature-based visual annotations

The visual annotations used in our visualization tool distinguish cells of different states by colors, which provide salient visual hints to users. The tool tracks and establishes the correspondence of cells across different time points to allow coherent cell annotations over time. In time-lapse live-cell imaging, cell tracking and lineage construction is a complex problem that requires a global context for long-term inter-relationship across various frames [12]. Because neurons in our target dataset do not undergo cell division, we choose Kernelized Correlation Filter (KCF) [13] to track cell movements. Individual KCF trackers, one for each cell, update the positions of the bounding boxes frame-by-frame following the paths of cell movement. The initial bounding boxes in the first frame are labeled manually. Expert biologists have visually verified the correctness of both the initial bounding boxes and the tracking results.

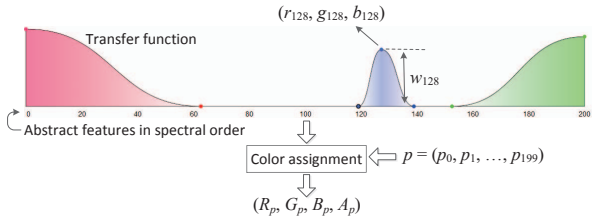


Fig. 1. After extracting 200 deep features, we assign each pixel p a color (R_p, G_p, B_p, A_p) based on the user-defined transfer function, which can be easily modified after the 200 features are spectrally ordered along the x-axis.

We now describe a user-mediated color assignment scheme (Fig. 1) to convert the 200 deep features, extracted from a trained CNN, into the colors used in our visualization. In the following we use $(p_0, p_1, \cdots, p_{199})$ to denote the L_2-normalized features extracted for a pixel p. Each deep feature p_i is assigned a weight w_i, whose value is determined by the curve given the control points. Each control point is associated with a color assigned by users. After interpolation the curve therefore creates a transfer function that defines a mapping from i to the corresponding color (r_i, g_i, b_i) and weight w_i. The color assigned to the pixel p, denoted by (R_p, G_p, B_p), is generated by a weighted average of (r_i, g_i, b_i) using $p_i w_i$ as the weight (Equation 1, G_p and B_p are calculated similarly). The i-th feature will contribute more to the aggregated color of p when p_i and w_i are larger. The opacity assigned to the pixel p, denoted by A_p, is the sum of the weights used in color aggregation. The opacity A_p represents the importance of the pixel p in comparison with other pixels.

$$R_p = \sum_{i=0}^{199} (p_i w_i) r_i \quad (1) \qquad A_p = \sum_{i=0}^{199} p_i w_i \quad (2)$$

Finally, we calculate the color of a bounding box by blending the colors of the pixels inside of the bounding box with respect to their opacities. By augmenting the bounding boxes with the calculated composite colors, we can effectively annotate cells by their characteristics. For example, the transfer function in Fig. 1 leads to the annotations in Fig. 2, in which the change in color matches the change in the appearance of a cell that died between frames 49 and 54.

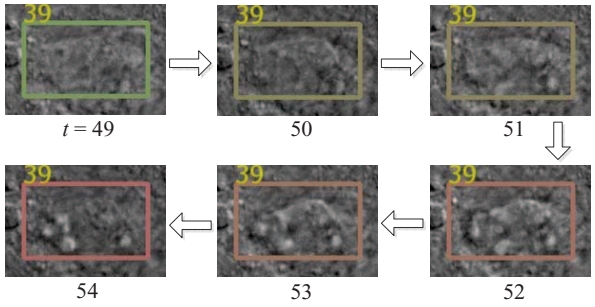


Fig. 2. The change in color (from green to red) of cell 39 from frame 49 to frame 54 suggests that the cell is dying.

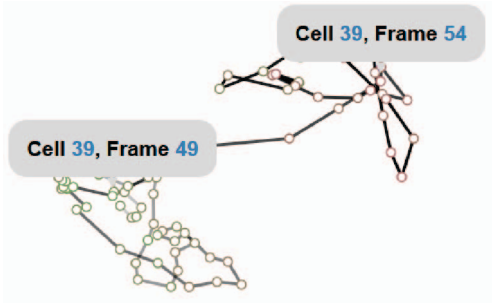
The color assignment scheme requires all 200 features to update the assigned color when the transfer function changes. This requirement imposes a huge memory burden. For example, we would need 69 gigabytes of memory to load the features for all the 86 time points in our dataset. We could assign colors offline but that would compromise interactivity, which we consider crucial in the visual reasoning process. As a result, we apply vector quantization using an incremental k-means algorithm [14] to create a codebook with k codewords (the centroids of clusters). Throughout this work, we use $k = 256$ such that the indices of codewords are one byte in size.

Because users can assign colors and weights to features in groups based on their proximity along the x-axis in the design widget, a proper ordering of features will reduce the complexity of transfer functions. We address this usability issue by spectral ordering [15] features along the x-axis in Fig. 1 based on pairwise feature similarity such that similar features are closer to each other. This ordering allows users to design suitable transfer functions with far fewer control points, thereby accelerating the design of transfer functions.

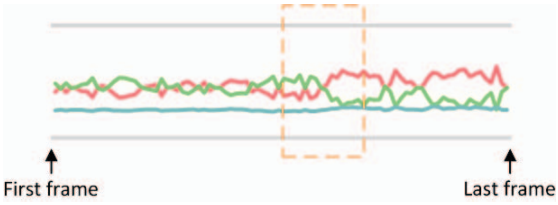
The spectral ordering of features consists of the following steps. First, a feature-to-feature similarity matrix A is created by assessing pairwise similarity using the inner product of two feature vectors. Second, we calculate a diagonal matrix D, in which the diagonal elements $D(i, i)$ is the sum of the i-th row vector of A. Finally, after calculating A and D, we order the features using the Fiedler vector, which is the eigenvector corresponding to the second smallest eigenvalue of the Laplacian matrix $L = D - A$. After re-ordering the feature, the proximity of two features along the x-axis approximately corresponds to their similarity.

2.4. Summarization of cell changes

The visualization we have presented in Fig. 2 is still a frame-by-frame playback of cells between two time points. Here we assume that cell positions are unrelated to their states and use the position in screen space to represent the trajectory of change in cell states, leading to a visualization that shows a summarization over a period of time (Fig. 3).



(a) Trajectory of a cell



(b) Color change over time

Fig. 3. The state change of a cell is visualized as a segmented line. The change in color from green to red indicates a state change of cell 39 from live to dead between frame 49 and 54.

The 2D layout of cells is determined as follows. We use the color histogram of pixels enclosed in the bounding box to represent a cell at a specific time point. The dissimilarity between a pair of cells is assessed using the Earth mover's distance (EMD) of the corresponding pair of color histograms. We use the average EMD of three pairs of histograms (i.e. R, G, and B) as the final dissimilarity between pairs of cells. Given the pairwise dissimilarity matrix, we can now create a 2D layout of cells such that similar (or dissimilar) cells are closer (or farther) through multidimensional scaling [16]. After obtaining the 2D coordinates of cells, the trajectory of a cell is represented as a segmented line formed by segments connecting the cells in adjacent frames (Fig. 3a). The grayscale levels (white to black) of segments represent the progress of time. For example, Fig. 3a shows the trajectory of cell 39, which starts dying around frame 49 (c.f. Fig. 2). Between frame 49 and 54 the location of cell 39 in the 2D layout changes from the bottom left to the top right while its color

changes from green to red. Based on this visual representation, we build an interactive visualization tool that allows users to select cells with similar trajectory profiles. The tool also shows the change in color over time (Fig. 3b). From left to right, color lines depict the changes in red, green, and blue over time for cell 39. During the time period within the orange box, the red line increases while the green line decreases.

3. EXPERIMENTS AND RESULTS

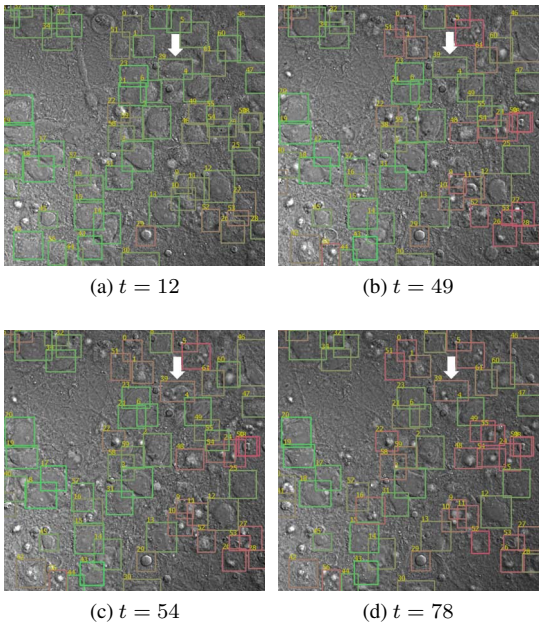
The color-based annotations in most cases are consistent with the changes in cell states. Fig. 4a–4d show the augmented frames at $t = 12$, 49, 54, and 78, respectively. Users can quickly identify which cells are dead by the color of their bounding boxes. Fig. 5 shows the trajectories of cells 3, 38, and 39. Because the color assigned to cell 3 is always green, users can easily conclude that cell 3 stays alive from the first time point to the last. Both cells 38 and 39 died but they have distinct trajectory profiles. By looking at the color lines (top of Fig. 5), we observe that the green line of cell 38 stays low and never grows back again once it starts decreasing. The green line of cell 39, however, shows a jagged pattern. After having a closer look in Fig. 2 and Fig. 4, we believe the fix-sized bounding boxes returned by KCF caused the jagged pattern because the bounding box no longer tightly fits cell 39 after it died. As a result, the bounding box inevitably includes many pixels that belong to the background, thereby introducing noise into our color assignment scheme.

4. CONCLUSIONS

We present a method that uses 3D CNN to detect and depict temporal changes in cell states in live-cell images. Based on the abstract features derived directly from the data, we build a visual analytics tool that allows users to create comprehensive color-based annotations of cells. Our tool also creates a summarization-based visualization that shows the change of a cell over time as a segmented line. This static 2D layout allows users to compare cells or search for cells with similar trajectory profiles. The promising results show that our tool can improve the understanding of complex time-varying datasets. We plan to work with biomedical experts and apply the same analyses to other live-cell image datasets. We also plan to extend our deep-learning-assisted techniques for multi-dimensional volume visualization applications [17, 18].

5. ACKNOWLEDGEMENTS

This work has been supported in part by the NSF Grants 14-29404, 15-64212, NIH RO1 NS052568, NIST Grant #70NANB15H329, the State of Maryland's MPower initiative, and the NVIDIA CUDA Center of Excellence. Any opinions, findings, conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the research sponsors.



(a) $t = 12$ (b) $t = 49$

(c) $t = 54$ (d) $t = 78$

Fig. 4. In this example, most of the cells are alive (green) at time point 12. They die (red) gradually in subsequent time points ($t = 49$, 54, and 78). Cell 39 (white arrow), which changes color from green to red, is dead sometime during frame 49 and 54.

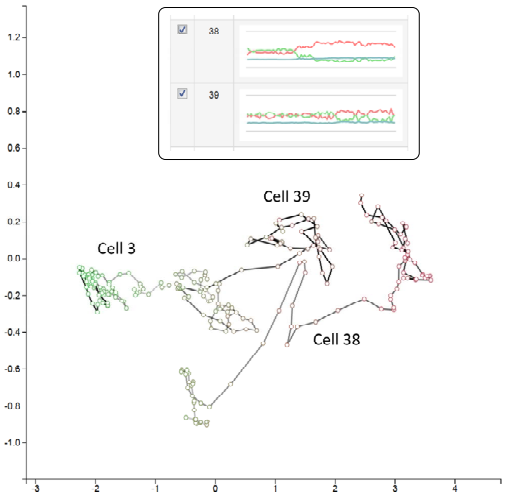


Fig. 5. Our visualization allows users to select and compare cell trajectories without consulting different frames. This example shows three cells with distinct trajectories. Cell 3 stays alive whereas cells 38 and 39 died at specific frames, hinted by the drop in color green (and the growth in color red).

6. REFERENCES

[1] Bogdan A. Stoica, David J. Loane, Zaorui Zhao, Shruti V. Kabadi, Marie Hanscom, Kimberly R. Byrnes, and Alan I. Faden, "PARP-1 inhibition attenuates neuronal loss, microglia activation and neurological deficits after traumatic brain injury," *Journal of Neurotrauma*, vol. 31, no. 8, pp. 758–772, Apr. 2014.

[2] Dan Ciresan, Alessandro Giusti, Luca M. Gambardella, and Jürgen Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Proceedings of the Annual Conference on Advances in Neural Information Processing Systems*, 2012, pp. 2843–2851.

[3] Dan C. Cireşan, Alessandro Giusti, Luca M. Gambardella, and Jürgen Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2013, pp. 411–418.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the Annual Conference on Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[5] Cheuk Yiu Ip, Amitabh Varshney, and Joseph JaJa, "Hierarchical exploration of volumes using multilevel segmentation of the intensity-gradient histograms," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2355–2363, 2012.

[6] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, Jan. 2013.

[7] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proceedings of IEEE International Conference on Computer Vision*, 2015, pp. 4489–4497.

[8] Robert Patro, Cheuk Yiu Ip, Sujal Bista, Samuel S. Cho, D. Thirumalai, and Amitabh Varshney, "MDMap: A system for data-driven layout and exploration of molecular dynamics simulations," in *Proceedings of IEEE Symposium on Biological Data Visualization*, 2011, pp. 111–118.

[9] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of International Conference on Machine Learning*, 2015.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.

[11] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the ACM International Conference on Multimedia*, 2014, pp. 675–678.

[12] Kang Li, Mei Chen, and Takeo Kanade, "Cell population tracking and lineage construction with spatiotemporal context," in *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention*. Oct. 2007, pp. 295–302, Springer, Berlin, Heidelberg.

[13] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[14] David Sculley, "Web-scale k-means clustering," in *Proceedings of the International Conference on World Wide Web*, 2010, pp. 1177–1178.

[15] Bojan Mohar, "Laplace eigenvalues of graphs—a survey," *Discrete Mathematics*, vol. 109, no. 1, pp. 171–183, Nov. 1992.

[16] Joseph B. Kruskal and Myron Wish, *Multidimensional Scaling*, Sage Publications, Beverly Hills, 1978.

[17] Sujal Bista, Jiachen Zhuo, Rao P. Gullapalli, and Amitabh Varshney, "Visualization of brain microstructure through spherical harmonics illumination of high fidelity spatio-angular fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2516–2525, Dec. 2014.

[18] Sujal Bista, Jiachen Zhuo, Rao P. Gullapalli, and Amitabh Varshney, "Visual knowledge discovery for diffusion kurtosis datasets of the human brain," in *Visualization and Processing of Higher Order Descriptors for Multi-Valued Data*, pp. 213–234. Springer, Cham, 2015.