JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 25, Number 8, 2018
Mary Ann Liebert, Inc.
Pp. 934–953
DOI: 10.1089/cmb.2018.0023

Computational Analysis of Cell Dynamics in Videos with Hierarchical-Pooled Deep-Convolutional Features

FENGQIAN PANG, HENG LI, YONGGANG SHI, and ZHIWEN LIU

ABSTRACT

Computational analysis of cellular appearance and its dynamics is used to investigate physiological properties of cells in biomedical research. In consideration of the great success of deep learning in video analysis, we first introduce two-stream convolutional networks (ConvNets) to automatically learn the biologically meaningful dynamics from raw live-cell videos. However, the two-stream ConvNets lack the ability to capture long-range video evolution. Therefore, a novel hierarchical pooling strategy is proposed to model the cell dynamics in a whole video, which is composed of trajectory pooling for short-term dynamics and rank pooling for long-range ones. Experimental results demonstrate that the proposed pipeline effectively captures the spatiotemporal dynamics from the raw live-cell videos and outperforms existing methods on our cell video database.

Keywords: cell dynamics, deep convolutional features, deep convolutional networks, hierarchical pooling.

1. INTRODUCTION

Characterizing various temporal dynamics of individual cells provides useful cell phenotypic information and is a powerful approach to modeling cell cycle stages, analyzing migratory phenotypes, and unraveling cellular response to physiological stimuli (Chechik and Koller, 2009; Spiller et al., 2010; Xiong and Iglesias, 2010; Zhong et al., 2012; Gordonov et al., 2016; Hu and Yang, 2016; Wang and Cong, 2016; Niederberger et al., 2015; Obayemi et al., 2016; Yuan et al., 2012; Zhong et al., 2016). Recently, the development of time-resolved cell imaging technology has not only enabled biologists to visualize cellular behaviors with more details but also brought great potential to analyze the live-cell dynamics based on image processing and machine learning (Dunkers et al., 2012; Li et al., 2015a; Wang et al., 2015b).

There are several kinds of cell dynamics in videos, such as cell morphology variation, nuclear deformation, and intracellular movement, applied to different applications. For example, subcellular motility is exploited as a complement of morphological features to distinguish live cells from the dead ones (Dunkers et al., 2012). Meanwhile, cell morphology dynamics is clustered with temporal constraint to model the mitosis procedure (Zhong et al., 2012; Gordonov et al., 2016). To encapsulate the cell dynamics in a whole video, it is not only needed to profile the local dynamics in video segments but also to fuse them into a

Department of Information and Electronics, Beijing Institute of Technology, Beijing, China.

video-wide representation (Zhong et al., 2012; Pang et al., 2015). Therefore, most of the process in the field of cell dynamics is associated with either of short-term local dynamics or long-term video-range ones.

The key to assessing short-term cell dynamics is to precisely characterize the cellular appearance and its change on consecutive frames. The shape parameters (such as volume, centroid, and circularity), Zernike moments, and radial distance can describe cellular morphology, and hence, their variation on several neighbor frames serves as the index of cell morphology dynamics (Xiong and Iglesias, 2010; An et al., 2012; Li et al., 2014, 2015b; Tsygankov et al., 2014; Kotyk et al., 2015; Alizadeh et al., 2016). Meanwhile, a spatial transformation is learned to warp one cellular contour to another, and the parameters of this shape transformation are defined as the measurement of cell deformation or applied to identify the variation modes of normal Hala nuclei (Rohde et al., 2008; Johnson et al., 2015). Besides the dynamics of cytoplasmic streaming is modeled by constructing the movement field, whose horizontal and vertical average velocities are used to classify the cells with different dynamic modes (Huang et al., 2013; Pang et al., 2015; Kaviani et al., 2016).

In recent years, deep learning techniques have achieved great success in many image-based tasks (Krizhevsky et al., 2012; Simonyan and Zissermann, 2015; Szegedy et al., 2015), and there have been a lot of attempts to develop deep architectures for cell-related research, such as cell segmentation (Sadanandan et al., 2016) and classification (Chen et al., 2016). However, these existing cell-related deep neural networks may not be directly extended to learning cell dynamics because of the existing temporal domain. In this article, we introduce two-stream convolutional networks (ConvNets) to automatically learn the convolutional feature maps with meaningful dynamics from raw live-cell videos. Then, a kind of temporal pooling strategy, trajectory pooling, is applied to the discriminative convolutional feature maps instead of the original video, and it will further enhance the capability of capturing the complex cell dynamic information over a short time.

Another line of research on cell dynamics attempts to capture the long-term cell dynamics, because a lot of intracellular signaling processes may take minutes or even hours (Spiller et al., 2010). The straightforward method aggregates the short-term dynamic features by the concatenation or accumulation strategy (An et al., 2012; Huang et al., 2013; Li et al., 2015b; Myers and Rabiner, 1981). To preserve more temporal structures in live-cell videos, temporal bag of words (TBoW) is introduced to learn a visual codebook based on frame-level dynamic features and encode the long-term cell dynamics of a particular video as a histogram corresponding to the word frequency of the visual codebook (Pang et al., 2015). Similarly, hidden Markov models (HMM) can profile the cell shape dynamics in time series as a temporal sequence of predefined morphological states, which are regarded as the words in visual codebook (Held et al., 2010; Zhong et al., 2012; Gordonov et al., 2016). Therefore, HMM is used to annotate the cellular phases during cell division or mitosis in time-lapsed movies (Held et al., 2010; Zhong et al., 2012). HMM also condenses the temporal dynamics in live-cell videos into a simpler representation, sequences of morphological states, for cellular response-based drug classification (Gordonov et al., 2016). However, either TBoW or HMM needs to specify prior information about the codebook size or cellular states (Hasan et al., 2016).

In comparison with TBoW and HMM, rank pooling captures the latent structure of the whole video by learning a pooling function via rank machines, which optimize a max-margin objective under the frame order constraints (Fernando et al., 2017). Rank pooling is not only independent of prior knowledge but also a more flexible alternative to encapsulating the video-wide temporal information. Therefore, rank pooling may be more suitable to capture the long-range cell dynamics, and is incorporated into our framework to learn the temporal evolution of cellular deep convolutional features aggregated by trajectory pooling.

In this article, we propose a novel framework for analyzing cell dynamics as shown in Figure 1. This framework shares the merits of both the deep learning technique and the hierarchical temporal pooling method. On the one hand, pretrained ConvNets embed the cell dynamics in video segments into convolutional feature maps with biological significance (for details see Section 3). On the other hand, the proposed hierarchical temporal pooling strategy encompasses the hierarchy of Figure 1 (corresponding to Section 4). At the short-term level, trajectory pooling first computes trajectories from live-cell videos and then aggregates the convolutional feature maps along these trajectories. It is able to further boost the capability of capturing the dynamics over a short time for the above ConvNets. Meanwhile, the long-range cell dynamics can be modeled by rank pooling. It learns a pooling function to rank the extracted short-term features of a live-cell video in chronological order, and the parameters of the ranking function are used as the video-wide dynamic features for the whole live-cell videos.

Overall, there are three key advantages to this work. First, the deep neural network is extended from the applications based on cell image processing to the tasks of cell video processing, for example, the analysis

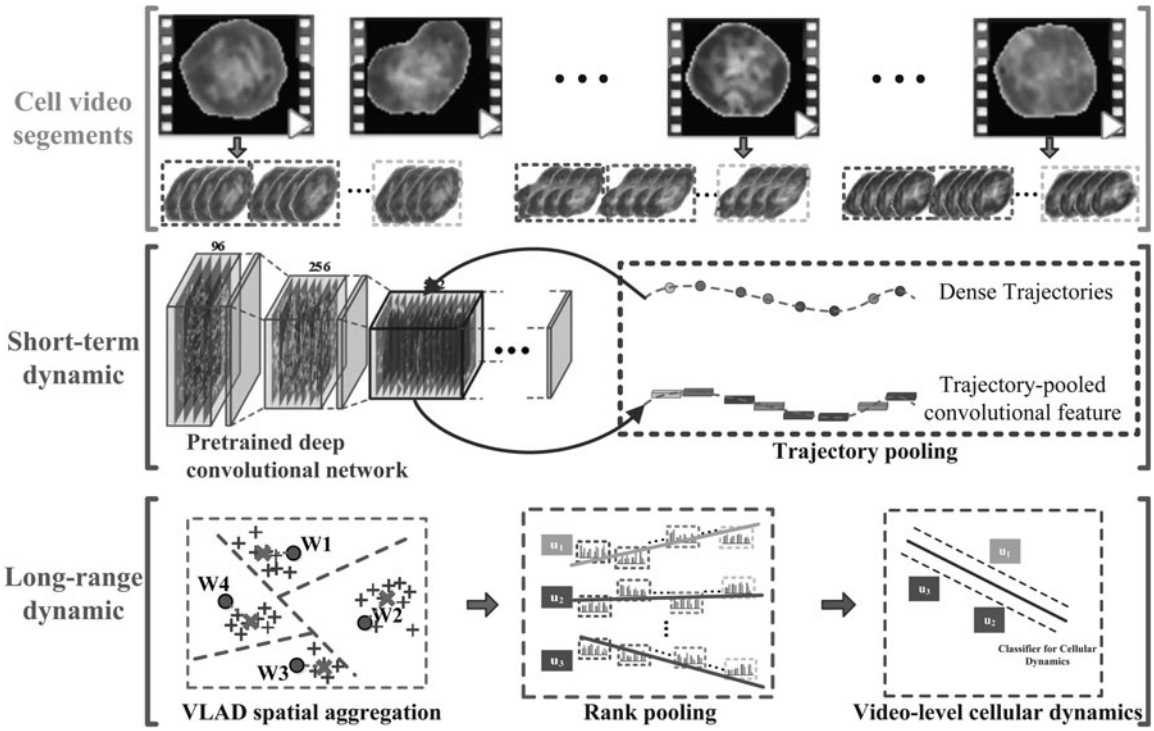

FIG. 1. Hierarchical structure of the proposed temporal pooling strategy to learn cell dynamics. At short-term level, trajectory pooling captures cell dynamics from the convolutional feature maps of ConvNets. Meanwhile, rank pooling can model the long-range cell dynamics by learning the chronological order of the short-term dynamic features. ConvNets, convolutional networks.

of cell dynamics. The second advantage of this method is that a proposed hierarchical temporal pooling can facilitate the capability of encoding cell dynamics at both short-term level and long-range level for the deep neural network. The third advantage is a remarkable performance of the proposed framework and its convenient extension to various cell investigations.

2. MATERIALS AND DATA SET

This section mainly describes the live-cell video data set for evaluating the utility of the proposed method. The data set comprised 120 video clips of lymphocytes from male mice undergoing skin transplantation. These video clips are further annotated into four categories by three experts according to the variation type of lymphocytes. Thus, the details about cell video acquisition and cell data annotation are explained in Sections 2.1 and 2.2, separately.

2.1. Materials and data acquisition

The experimental data are acquired with skin transplant surgery for male mice in collaboration with Beijing You An Hospital. During the skin transplantation, there are two kinds of male mice, healthy C57BL/6 and healthy Balb/C male mice, to be used as donors and hosts, respectively. Both kinds of mice were 6 to 8 weeks old with the weight of 20 to 22 g and were obtained from Graduate School of the Academy of Military Medical Sciences (Beijing).

The skin transplant process could be conducted in the following three stages. First, the transplanted skin was displaced in the zone near the back of donors after they were decapitated and their torso hair was shaved off. Then the skin was further cut into several $1.5cm \times 1.5cm$ regular squares and preserved in sterile phosphate-buffered saline. Second, the hosts were injected sodium pentobarbital ($70\,\mu g/g$ *of body weight*) for anesthetization, and their back hair was also shaved off. The surgical area was cleaned with alcohol, disinfected with iodine, and then removed. In there, the transplanted skin from the donors was

positioned, covered with vaseline gauze, and fixed with a band-aid. Finally, the hosts are raised individually in sterile cages in the ventilated environment for seven days.

On the seventh day after skin transplantation, peripheral blood smears were prepared from the blood samples collected from the tails of mice. Each blood smear was observed with phase-contrast microscopy (Olympus BX51, 0.3μ resolution, 16×1000 magnification). Each time only one target lymphocyte was manually positioned in the center of the field, and its deformation or dynamics was recorded in a video clip (20–30 seconds, 25 frames per second). Then, a quality control step was conducted beforehand to make sure there is no overlap and trajectory cross between the lymphocyte and red blood cells. These video clips of lymphocytes were collected in the data set,* and several samples are shown in Figure 2.

2.2. Data preprocessing and annotation

Automated image preprocessing, cell segmentation (Li and Acton, 2007), and cell tracking (Li et al., 2014) were performed for the acquired experimental data. Then, manual validation is exploited to eliminate the ambiguity of cell segmentation by human eye if necessary. As we mainly focus on the variation of cellular appearance, the preprocessed data can serve as a starting point to model the cellular dynamics with various methods.

The preprocessed data were equally divided into four categories, the labels of which were denoted as normal, slight activation, moderate activation, and drastic activation, separately. This annotation process was performed by three experts. Specifically, the experts observed the video clips of lymphocytes and then annotated each of them with a label according to the dynamic modes of lymphocytes therein. To enhance the confidence of annotation, the annotations were not only performed by three experts but also performed by the same expert on two different days.

Therefore, the labels obtained an additional confidence level with respect to the consistency of annotations: (1) those labels that agreed in all six annotation processes were considered suitable for the analysis of cell dynamics and relatively *easy* for this classification task; (2) when the number of annotations in the same category was more than four, these labels could also be used in the classification task, but the corresponding samples were *difficult* for the proposed or other mainstreaming approaches; and (3) the rest of the video clips were removed from the data set. At last, each category in dataset contained 30 videoclips, and the *difficult* ones therein were no more than 20%. In this article, our experiments are based on both *easy* and *difficult* samples.

3. DEEP CONVOLUTIONAL FEATURE MAPS FOR CELL DYNAMICS

In this section, the deep ConvNet is regarded as a generic feature extractor for cell dynamics. First, we present the architecture of the ConvNet and its pretrained model used in our framework. Then, we show the characteristics of the feature maps from different convolutional layers. Finally, the representation of cell dynamics can be visualized based on the aforementioned deep learning technology. The cell dynamics based on deep learning can be visualized, and consistency with the data annotations can be reflected in some degree.

3.1. Pretrained deep convolutional networks

As two-stream ConvNets have been successfully applied to action recognition and activity recognition, its two-stream pipeline also becomes one of the popular deep learning frameworks in video analysis field (Simonyan and Zisserman, 2014; Feichtenhofer et al., 2016; Wang et al., 2015c; Zhao et al., 2015). This two-stream pipeline assumes that human action or activity can be decomposed into temporal and spatial components. The temporal part captures the motion or dynamic information, which is an intrinsic property of action or activity recognition. Meanwhile, the spatial part carries static appearance cues, such as human dressing or postures and information about scenes, and is beneficial to the recognition tasks. Therefore, the

*Several examples are available at http://isip.bit.edu.cn/kyxz/xzlw/77051.htm

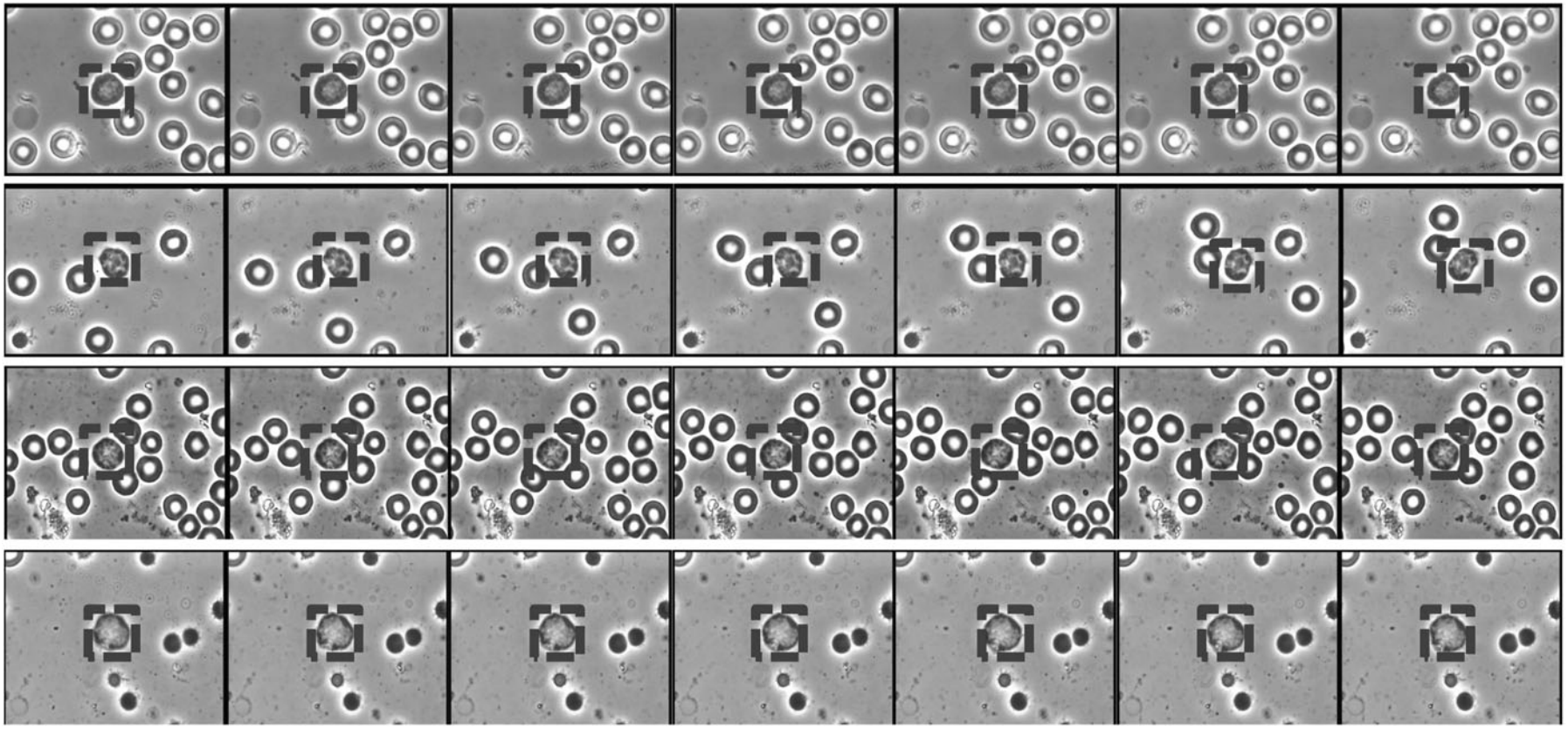

FIG. 2. Samples from live-cell data sets arranged in rows. The frames in different columns are extracted from certain videos at the fixed time interval of 3 seconds. Only one target lymphocyte (in the blue dashed box) is observed once, and there are no overlap and trajectory cross between the lymphocyte and red blood cells.

two-stream pipeline is implemented by two separate neural networks, that is, a temporal net and a spatial net individually learn the dynamic information and static appearance cues.

However, the research of cell dynamics is usually about the specific type of cells under nearly the same condition. As a result, the spatial part, for example, cellular morphology, cannot be proved to have a direct relationship with the variation type of cells. In this article, only the temporal net is introduced from the two-stream framework provided by Wang et al. (2015c). The temporal ConvNet starts with the dense optical flow field constructed with TVL1 algorithm (Zach et al., 2007). The optical flow field is calculated between pairs of consecutive frames, and the values therein are further normalized into the integer range [0, 255], just like images.

3.1.1. Temporal convolutional network. The aforementioned optical flow fields are stacked into $224 \times 224 \times 2F$ (F is the number of stacked flows) volumes as the input. The temporal net is composed of five convolutional layers, three pooling layers, and two fully connected layers (as shown in Fig. 3). Furthermore, some other details of its architecture are summarized in Table 1. The first and second convolutional layers individually have the convolutional size 7×7 and 5×5, but both have the convolutional stride 2 and postprocessing of local response normalization. Meanwhile, the following convolutional layers share the same convolutional size (3×3) and convolutional stride (1). Then, there are three pooling layers (pooling window 3×3) cascaded with first, second, and fifth convolutional layers, respectively. Finally, the last three layers sequentially comprise two fully connected layers regularized by dropout operation, and a SoftMax layer served as a classifier.

3.1.2. Pretrained model of temporal convolutional network. Since our work cannot satisfy the excessive requirement of large numbers of annotated samples for training, we use the pretrained temporal ConvNet reported in the literature (Wang et al., 2015c). Although this model contains the spatial net and the temporal net, we only focus on the temporal one. The number of stacked flows F is set to 10, and therefore, the input is a $224 \times 224 \times 20$ subvolume that is randomly cropped and flipped from the training videos. As the samples of UCF-101 data set are not very sufficient for training, we select high dropout parameters (0.9 for full6 layer and 0.8 for full7 layer) to improve the generalization of the pretrained model. The training process needs about 90K iterations: the learning rate is initialized as 10^{-2} before 50K iterations, then reduced to 10^{-3} during 50K–70K iterations, and finally decreases to 10^{-4} until training is stopped.

3.2. Convolutional feature maps

When pretrained temporal ConvNet learns a new testing sample, particularly a live-cell video in our work, the raw video stream flows into its bottom-up architectures. The dynamic information in the video is represented as deep features at several levels of granularity by different layers. The shallower convolutional

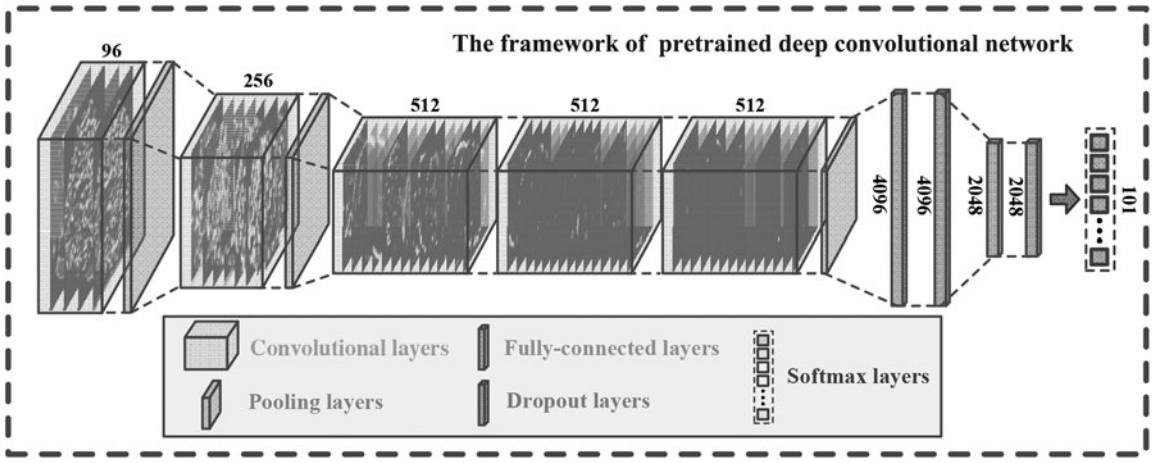

FIG. 3. Illustration of the pretrained temporal ConvNet. From bottom to top, this pretrained model mainly consists of several couples of alternating convolutional and pooling layers, two full-connected layers regularized by dropout operation (also regarded as dropout layer), and a SoftMax layer for classification.

Table 1. Temporal Convolutional Network Architectures

Layer	*Conv1*	*Conv2*	*Conv3*	*Conv4*	*Conv5*	*Full6*	*Full7*	*SoftMax*
Size	7×7	5×5	3×3	3×3	3×3	—	—	—
Stride	2	2	1	1	1	—	—	—
Channel	96	256	512	512	512	4096	2048	101
Normalized	LRN	LRN	—	—	—	—	—	—
Pool	3×3	3×3	—	—	3×3	—	—	—

LRN, local reponse normalized.

layers contain local characteristics of motion, while deeper convolutional layers capture the global dynamic patterns. As the higher layers obtain more abstracted and discriminative features, the fully connected layers can express high-level concepts and are naturally applied in classification and recognition tasks (Simonyan and Zisserman, 2014). Recently, convolutional layers are more widely utilized because of the advantage that they can preserve the spatial structures (Wang et al., 2015c; Zhao et al., 2015). And this advantage makes convolutional feature maps more compatible with some hand-crafted methods, for example, trajectory pooling.

Therefore, we will exploit the convolutional feature maps from convolutional layers as dynamic features. Given a video V, x, y, and t denote the horizontal, vertical, and temporal positions in videos, separately. $C_m^t \in \mathbb{R}^{H_m \times W_m \times N_m}$ denotes the m-th convolutional feature map of temporal net on frame t, and H_m, W_m, and N_m stand for its height, width, and the number of channels, respectively. Then, spatiotemporal normalization and channel normalization are applied on these chosen convolutional feature maps (Wang et al., 2015c). Note that $\bar{C}_m^t$ stands for the m-th normalized convolutional feature map on frame t. Assume that u and v individually correspond to the horizontal and vertical positions in convolutional feature maps. $\bar{C}_m^t(u, v)$ can express a vector that concatenates the values in each channel of the feature map at position (u, v).

Intuitively, the feature maps from different layers describe the visual content at different levels. Consequently, the combination of convolutional feature maps is further exploited to boost the discrimination of pretrained temporal ConvNet. In detail, we choose the second and the third convolutional layers for our task of cell dynamic analysis based on the experimental results in Section 5.3.

3.3. Visualization of the learned feature maps

In this section, we demonstrate a visual inspection of what these convolutional layers learn. We first randomly choose one sample from both the normal activation category and the drastic activation one. Then, we put these two lymphocyte video clips into the pretrained network architectures. As a result, we obtain their feature maps from different convolutional layers, which correspond to the normal activation in Figure 4a and the drastic activation in Figure 4b. When comparing the *Conv*1 feature maps in the first row with the feature maps in other rows, it is apparent that *Conv*1 feature maps contain more subtile structures, standing for the local motion of cell dynamics. Therefore, this fact proves that shallower convolutional layers are inclined to capture local dynamic patterns to some extent. Between Figure 4a and b, the feature maps in the second, third, and fourth rows possess more discrimination, and the most discriminative feature maps are from *Conv*2. In the last row, we can hardly distinguish whether the *Conv*5 feature maps correspond to the normal activation or the drastic activation. This illustrates that cell dynamics can be represented by the shallower convolutional layers in deep architectures for action recognition.

4. HIERARCHICAL POOLING ON CONVOLUTIONAL FEATURE MAPS

In this section, we describe our hierarchical pooling strategy for convolutional feature maps. In the beginning, we extract the dense trajectories from short-term video segments and utilize these trajectories to pool the corresponding convolutional feature maps. Consequently, we can obtain local trajectory-pooled descriptors for the aforementioned short-term video segments. To further aggregate the spatially local dynamic descriptors into a global representation, vector of locally aggregated descriptors (VLADs) method is introduced. Finally, we present the details of rank pooling and use it to capture the long-range dynamics.

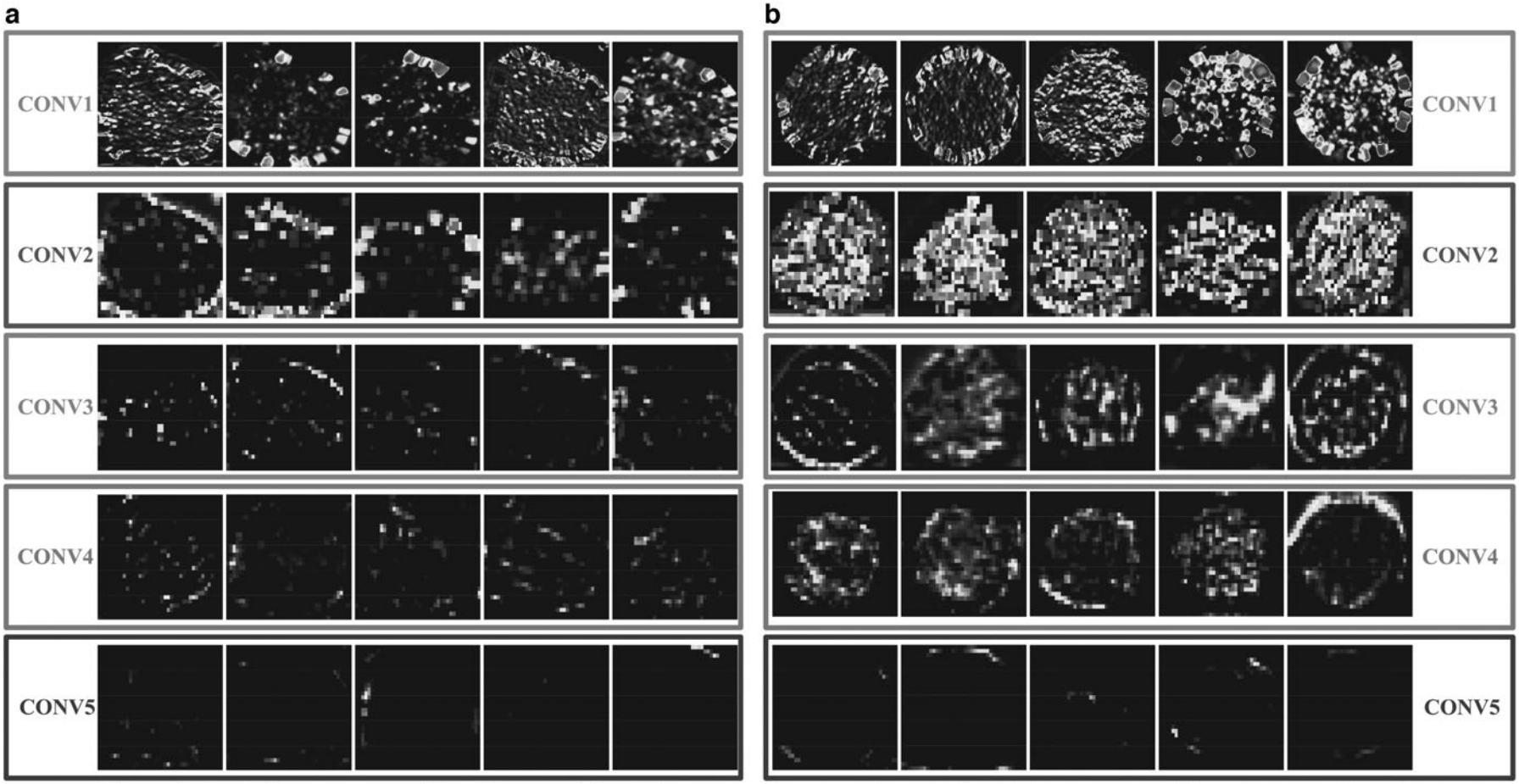

FIG. 4. Convolutional feature maps of temporal ConvNet learned from the video clips of the data set. From the first row to the fifth row, we show the feature maps from different convolutional layers (*Conv1–Conv5*). **(a)** and **(b)** correspond to the convolutional feature maps of lymphocyte video clips in normal activation and drastic activation, respectively.

4.1. Trajectory pooling for short-term dynamics

To capture more accurate short-term dynamics, trajectory pooling is applied to the stacked convolutional feature maps. It can incorporate the trajectory information learned from raw videos based on dense trajectories[†] (Wang et al., 2013), as shown in Figure 5. Given a cell video V, we exploit the pretrained temporal ConvNet to learn its corresponding convolutional feature maps $\bar{C}_m^t$, which stand for the m-th normalized convolutional feature map on frame t.

For subsequent trajectory pooling on convolutional feature maps, we first compute dense trajectories from raw videos. A set of feature points is densely sampled on a grid with multiple spatial scales, in which the grid step size is 5 pixels and the number of levels in the pyramid is 5. The spatial scales corresponding to different levels are individually $1/2$, $1/\sqrt{2}$, 1, $\sqrt{2}$, 2 times of the video spatial scale. Besides, feature points in homogeneous areas have no structure information, so they cannot be tracked through the video and should be removed. Specifically, these points on the grid are eliminated, if the eigenvalues of the autocorrelation matrices are smaller than a threshold (Shi et al., 1994). At last, the rest of the feature points are tracked frame by frame in dense flow fields.

Suppose $\omega = (u_t, v_t)$ is a dense flow field between frame t and frame $t+1$. For a feature point $P_t = (x_t, y_t)$ at frame t can be tracked by median filtering with a 3×3 filter kernel M:

$$P_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M * \omega)|_{(\bar{x}_t,\ \bar{y}_t)}, \tag{1}$$

where * is convolutional operation and $(\bar{x}_t, \bar{y}_t)$ is the rounded position of (x_t, y_t). The point is tracked in the subsequent frames, denoted as P_{t+2}, P_{t+3}, ... Then, a trajectory is formed by concatenating these tracked points (P_t, P_{t+1}, P_{t+2}, ...). To overcome the drifting problem, the trajectory length N is set to 15 frames (the same parameter setting as literature (Wang et al., 2013). Therefore, given a video V, the extracted dense trajectories can be denoted as $\mathbb{T}(V) = \{T^1, T^2, \ldots, T^k, \ldots, T^K\}$, where T^k stands for the k-th trajectory with 15 points:

$$T^k = \{(x_1^k, y_1^k, t_1^k), \ldots, (x_{15}^k, y_{15}^k, t_{15}^k)\}. \tag{2}$$

Once dense trajectories are extracted, we will exploit them to pool the convolutional feature maps. Specifically, a series of tracked points from trajectory T^k is mapped to the corresponding normalized feature map set $\{\bar{C}_m^t | m \in [2, 3], t \in [t_1^k, \ldots, t_{15}^k]\}$, so that we can obtain a trajectory-pooled descriptor $\mathbf{TD}^k = \{TD_m^k | m = 2, 3\}$, and TD_m^ktherein can be denoted as

$$TD_m^k = \sum_{i=1}^{N} \bar{C}_m^{t_i^k}(x_i^k \times \alpha,\ y_i^k \times \alpha). \tag{3}$$

Here $\alpha = H_m^t / H_v = W_m^t / W_v$ is a mapping ratio, H_m^t and W_m^t are the height and width of the m-th convolutional feature maps, while H_v and W_v denote the height and width of video frames, respectively.

4.2. Spatial aggregation for trajectory-pooled descriptors

For a video segment, its dynamics can be captured by a variety of trajectory-pooled descriptors. Thus, the global dynamics of this video segment has to spatially aggregate these descriptors by feature encoding methods, or VLAD in particular. As shown in Figure 6, the video segments in the training data set and a testing video segment are transformed into trajectory-pooled descriptors, denoted as $\{\mathbf{TD}_{train}^1, \mathbf{TD}_{train}^2, \ldots, \}$ and $\{\mathbf{TD}_{test}^1, \mathbf{TD}_{test}^2 \ldots, \mathbf{TD}_{test}^q, \ldots\}$, respectively. The following pipeline of VLAD can be summarized into two phases. In the training phase, training descriptors $\{\mathbf{TD}_{train}^1, \mathbf{TD}_{train}^2, \ldots, \}$ are clustered into L coarse centers $\{\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_l, \boldsymbol{c}_L\}$ by K-means (Jegou et al., 2012). In the testing phase, testing descriptors $\{\mathbf{TD}_{test}^1, \mathbf{TD}_{test}^2 \ldots, \mathbf{TD}_{test}^q, \ldots\}$ are assigned to the L coarse centers. Then, we can obtain the difference vector v_l with respective to the l-th center c_l for these trajectory-pooled descriptors by the following:

$$v_l = \sum_{q:NN(\mathbf{TD}_{test}^q) = c_l} (\mathbf{TD}_{test}^q - c_l) \tag{4}$$

[†]In comparison with action recognition, there is usually no motion about the microscopy during the recording process of cell dynamics. Therefore, here we only use dense trajectories rather than improve dense trajectories, whose modifications are about taking camera motion into account (Wang *et al*., 2015a).

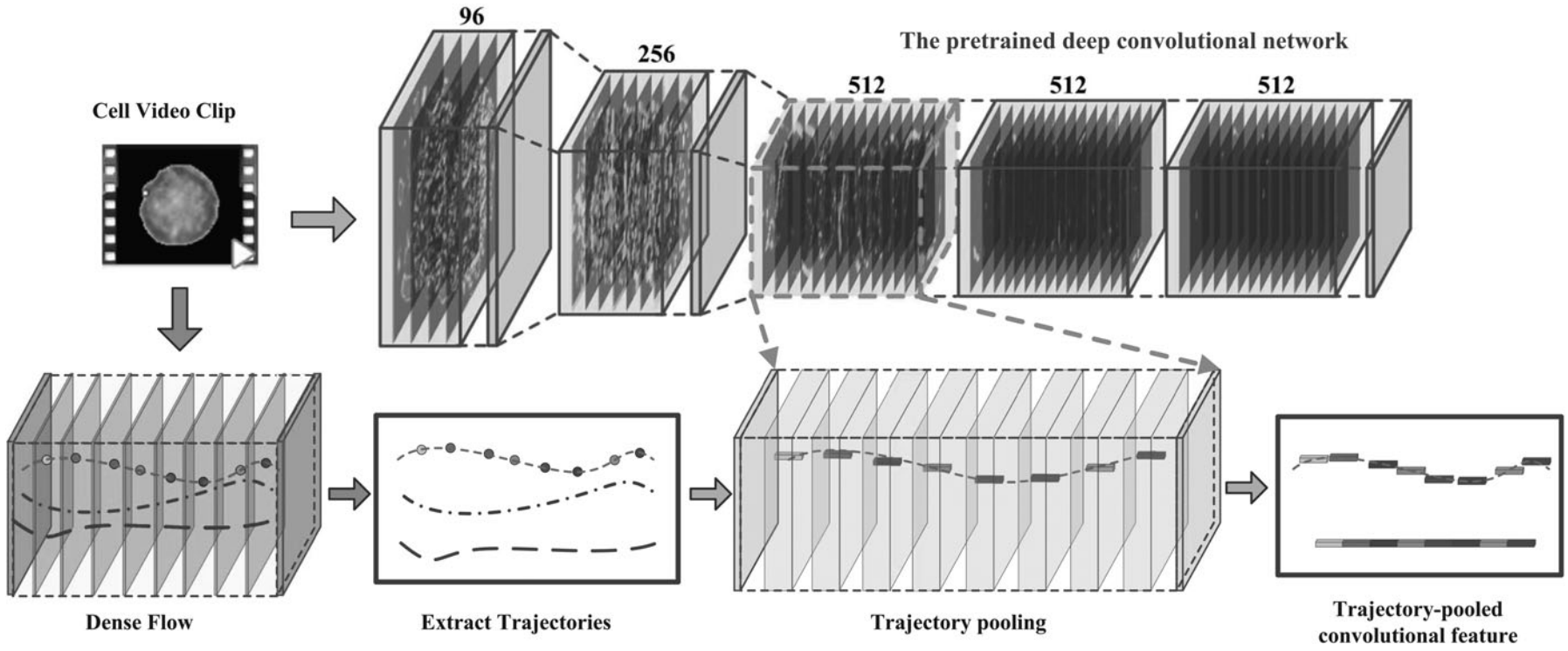

FIG. 5. Illustration of extracting trajectory-pooled descriptors based on convolutional feature maps. Given a cell video clip, its convolutional feature maps can be obtained by the pretrained temporal ConvNet. Then, we calculate the trajectories from the raw video clip. Afterward, we apply trajectory pooling on the convolutional feature maps to extract the trajectory-pooled descriptors.

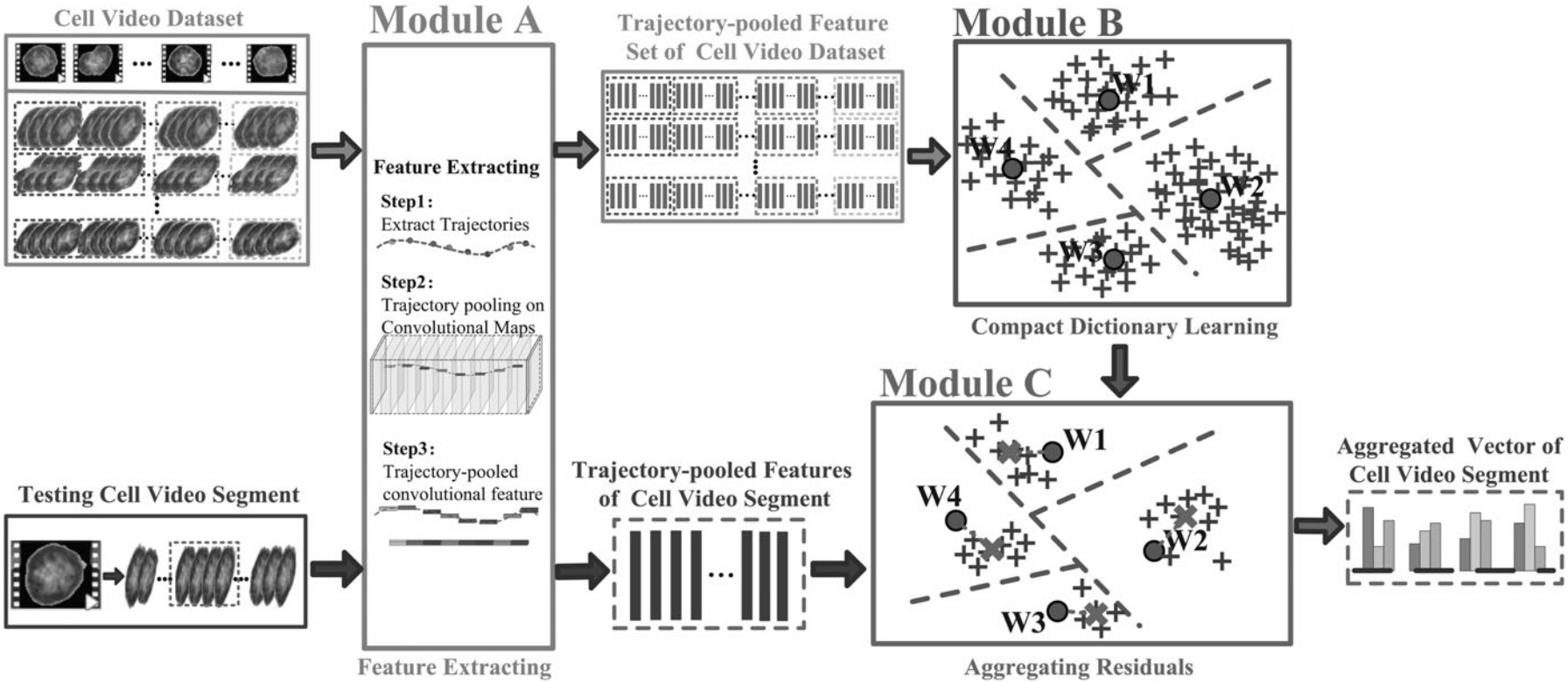

FIG. 6. Processing steps of VLAD-based spatial aggregation for trajectory-pooled descriptors. The module A is responsible for extracting trajectory-pooled descriptors for cell video data set or testing cell video segment. And the module B plays a part in generating a compact dictionary. Based on the compact dictionary, the module C encodes the trajectory-pooled vectors from the module A into a global VLAD, aggregating the cell dynamics in this video segment. VLAD, vector of locally aggregated descriptor.

where $NN(\mathbf{TD}_{test}^{q})$ indicates $\mathbf{TD}_{test}^{q}$'s nearest neighbors among L coarse centers. The VLAD encoding vector for trajectory-pooled descriptors $\mathbf{i}$ concatenates v_l over all the K centers with size $D\prime K$, and the post-processing uses the power and ℓ_2 normalization.

4.3. Rank pooling for long-range dynamics

One way to understand the video dynamics is to consider it as the relative ordering of the frames in the video, which can indicate the evolution of the frame appearance. Therefore, the learning-to-rank paradigm is introduced to fit the temporal variations in a video by ranking the frame-level features in chronological order (shown in Fig. 7). In details, a rank pooling function is optimized via ranking machines (Smola and Vapnik, 1997; Liu, 2009; Fernando et al., 2017) and its parameters serve as a robust new representation for video dynamics.

For a video sequence $\mathbf{V}$, we can denote the ordering of two frames therein as $\mathbf{f}_{t+1} > \mathbf{f}_t$ if $\mathbf{f}_{t+1}$ is followed by $\mathbf{f}_t$. As such, the frame order constraints of a video sequence, that is, $\mathbf{f}_n > \cdots > \mathbf{f}_t > \cdots > \mathbf{f}_1$, can be formulated as a series of pairwise order constraints according to the transitivity property of video frames. In other words, the constraints $\mathbf{f}_n > \cdots > \mathbf{f}_t > \cdots > \mathbf{f}_1$ can be equivalent to these pairwise constraints: $\mathbf{f}_n > \mathbf{f}_{n-1}$, $\cdots$, $\mathbf{f}_{t+1} > \mathbf{f}_t$, $\cdots$, $\mathbf{f}_2 > \mathbf{f}_1$. Hence, ranking the frame order in a video is transformed into a pairwise rank pooling problem for consecutive frames.

When using pairwise rank pooling to model the video dynamics, we exploit pairwise ranking machines to learn a linear function $\psi(\mathbf{f};\mathbf{u}) = \mathbf{u}^T \cdot \mathbf{f}$, where $\mathbf{u} \in \mathbb{R}^{\mathbb{D}}$ is the parametric vector and regarded as a new video representation. For each frame $\mathbf{f}_t$, its corresponding ranking score is $\psi(\mathbf{f}_t;\mathbf{u}) = \mathbf{u}^T \cdot \mathbf{f}_t$. The rank pooling tries to learn the parametric vector $\mathbf{u}$ to make all these constraints $\forall t_i,\ t_j,\ \mathbf{f}_{t_i} > \mathbf{f}_{t_j} \Leftrightarrow \mathbf{u}^T \cdot \mathbf{f}_{t_i} > \mathbf{u}^T \cdot \mathbf{f}_{t_j}$ satisfied. In this article, a live-cell video can be represented as a series of encoding vectors, $\{\boldsymbol{vtd_1},\ \ldots\ \boldsymbol{vtd_t},\ \ldots,\ \boldsymbol{vtd_n}\}$. We seek a direct mapping from $\boldsymbol{vtd_t}$ to its time variable t with the following objective function.

$$\psi(\boldsymbol{vtd}_t;\mathbf{u}) \mapsto t \tag{5}$$

$$\underset{\mathbf{u}}{arg\ min} \sum_{t}^{n} \left| t - \mathbf{u}^T \cdot \boldsymbol{vtd_t} \right|.$$

The direct mapping $\psi(\boldsymbol{vtd_t};\mathbf{u}) \mapsto t$ ensures that Eq. (5) satisfies the constraints $\forall t_i,\ t_j,\ \boldsymbol{vtd_{t_i}} > \boldsymbol{vtd_{t_j}} \Leftrightarrow \mathbf{u}^T \cdot \boldsymbol{vtd_{t_i}} > \mathbf{u}^T \cdot \boldsymbol{vtd_{t_j}}$ (Fernando et al., 2017). As a robust extension of Eq. (5), support vector regression (SVR) also learns a parametric vector $\mathbf{u}$ to represent how the short-term encoding vectors evolve over time in the live-cell video, or alternatively, the long-range cell dynamics.

5. EXPERIMENTS

In this section, we present a detailed experimental evaluation of our proposed framework based on the live-cell data set in Section 2. First, several exploration experiments are conducted to discuss the key parameters. Then, we evaluate the convolutional layers and pooling strategies. At last, our method is compared with other existing methods.

5.1. Experimental setup

Some important parameters for deep convolutional feature extraction and trajectory pooling are mentioned in Sections 3 and 4.1, while the rest is set as default parameters in the literatures (Wang et al., 2013, 2015c). The codebook size in VLAD method is discussed in the following Section 5.2. We use the SVR solver of Lib-linear toolkit[‡] (Fan et al., 2008) to learn rank pooling, and the penalty coefficient is set as a constant $C = 1$. In view of the high dimension of the final rank pooling vector, we apply linear support vector machine (SVM) as the classifier. It can be easily implemented with the SVM solver in Lib-linear toolkit, and the penalty coefficient is determined on the training set using fivefold cross-validation. As the proposed framework is a machine learning process, the data set should be divided into training set and test set. We use 30 random splits of the data, while considering 20 random video clips per class for training and

[‡]https://www.csie.ntu.edu.tw/cjlin/liblinear/

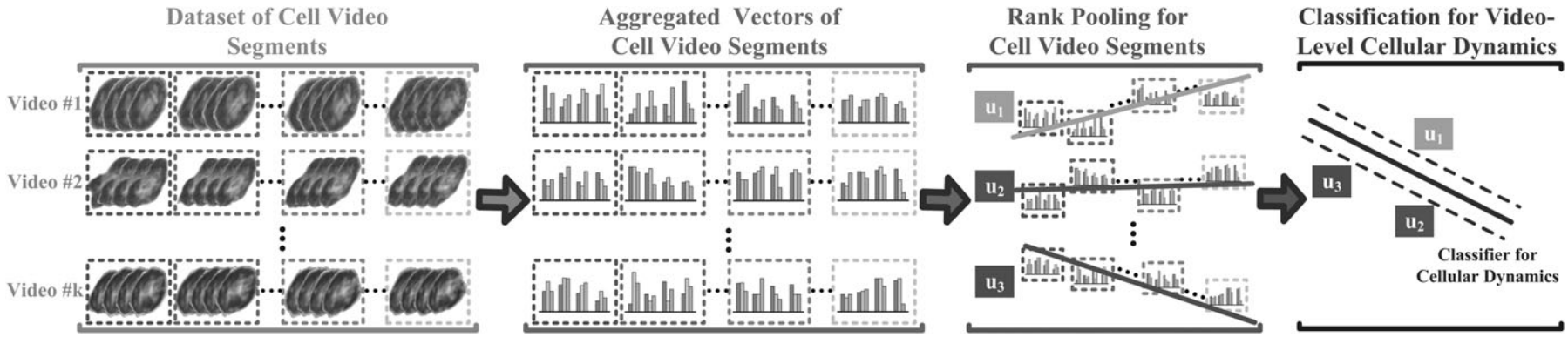

FIG. 7. Pipeline of rank pooling for long-range dynamics. First, we split each cell video clip into several video segments. Then, we generate the VLAD for each video segment as explained in Section 4.2. Afterward, based on RankSVR we learn the video representation u for each video. Finally, a classifier is set up in the u feature space.

the rest for testing. Therefore, the classification procedure is repeated for 30 times according to the splits, and we compute and report mean average precision (mAP), precision, recall, and F-score measures on the test set. The code of our proposed framework is provided at https://github.com/fengqian1989/HPDCF_code

5.2. Exploration experiments

5.2.1. Dimension reduction. The principal component analysis (PCA) method is usually introduced to decorrelate the local descriptors, such as dense trajectories or trajectory-pooled descriptors in this article. To specify the reduced dimension of trajectory-pooled descriptors, we explore PCA dimensions varying from 64 to 1014 on different convolutional layers and VLAD centers. In this exploration experiment, the mAP over all classes is calculated to evaluate the performance of the proposed method with different parameters. Figure 8a and f corresponds to the results of different VLAD centers (16 ∼ 512), separately. For each convolutional layer, we obtain the corresponding trajectory-pooled descriptor, which is denoted by the distinctive line type and maker. It can be inferred that each line in Figure 8 has an ascending tendency when the reduced dimension decreases. This illuminates that a lower dimension achieves a higher performance and dimension 64 is chosen as the default parameter in the rest of our experiments.

5.2.2. Vector of locally aggregated descriptor centers. The number of centers K is an important parameter of VLAD encoding. Intuitively, if the codebook size is too small, the histogram feature may lose the discriminative power, while if the codebook size is too large, the histograms from the same class may not possess enough similarity. Therefore, we perform trajectory pooling on five different convolutional

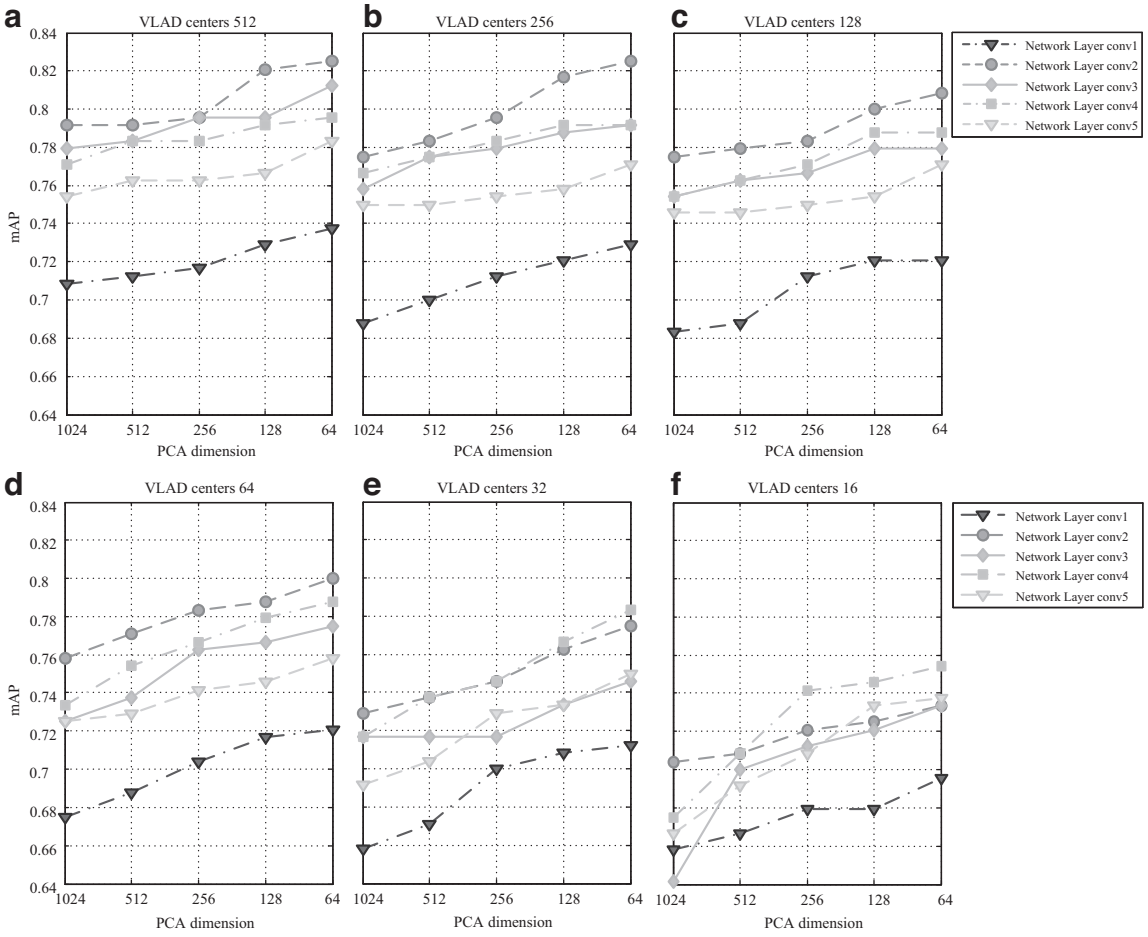

FIG. 8. Exploration of PCA dimension. From **(a)** to **(f)**, we present the results of different VLAD centers (512 ∼ 16). The different type of lines stand for different trajectory-pooled descriptors. For each line, the performance (mAP) is improved as PCA dimension decreases. mAP, mean average precision; PCA, principal component analysis.

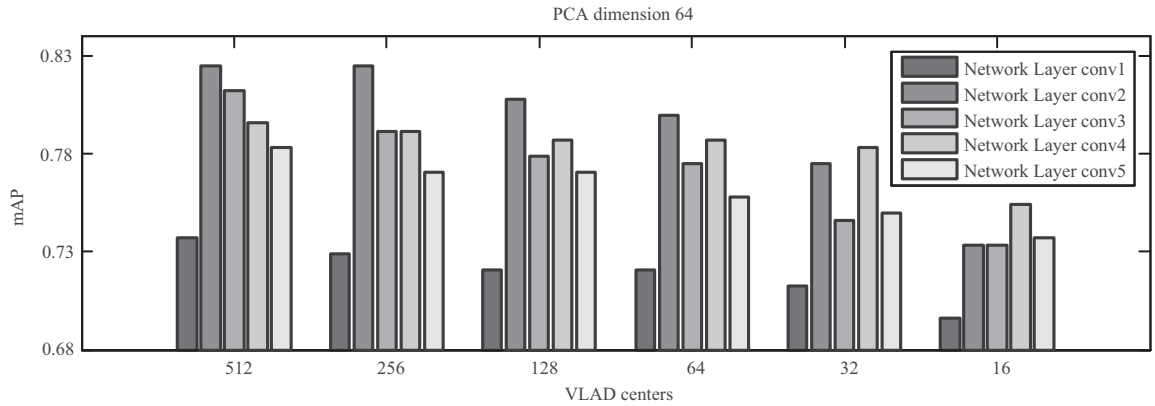

FIG. 9. mAP scores for different VLAD centers. On this group histogram, the bars with the same color belong to the same kind of trajectory-pooled descriptors from certain convolutional layers.

layers, and individually summarize the classification results (mAP) for different VLAD centers in the group histogram, as shown in Figure 9. On the group histogram, the bars with the identical color belong to the same kind of trajectory-pooled descriptors. When the convolutional layer *Conv*2 is chosen, the performance corresponding to $K = 256$ or 512 reaches the peak of the performance. In considering trajectory-pooled descriptors from all convolutional layers, $K = 512$ is specified as the default parameter in the remainder of this section.

5.3. Evaluation of different layers

In this section, we investigate the performance of different convolutional layers in temporal net on the live-cell video data set. For the pooling strategies, we only apply trajectory pooling on convolutional feature maps. The results are summarized in Figure 10a–e, and each subfigure corresponds to a table with respect to the parameter PCA dimension and VLAD centers. The different colors in Figure 10 stand for the different convolutional layers of ConvNets. Figure 10a presents which convolutional layers can reach the highest mAP scores for each parameter combination, while Figure 10b–e shows the convolutional layers with the second highest, the third highest, the second lowest, and the lowest mAP scores, respectively. As the color of *Conv*2 is in the majority in Figure 10a, it means that the *Conv*2 has the best performance among the convolutional layers. Similarly, the *Conv*3 and *Conv*4 achieve better performance than the *Conv*1 and *Conv*5. These experimental results are consistent with the visualization of convolutional layers in Section 3.3.

Then, we explore the performance of different combinations in *Conv*2, *Conv*3, and *Conv*4. Table 2 shows the performance of convolutional layers and their combinations with best parameter PCA dimension and VLAD center. From Table 2, we can see that *Conv*2 and its combinations achieve better performance than the others, which means it has higher discrimination. Among *Conv*2 and its combinations, *Conv*2 + *Conv*3 achieves the best performance 83.33. The combination of three layers decreases 1.25% performance from *Conv*2 + *Conv*3, because *Conv*4 is noncomplementary to *Conv*2 or *Conv*3 in some degree. Therefore, we choose the convolutional maps of *Conv*2 + *Conv*3 in the proposed framework.

5.4. Evaluation of pooling strategies

Based on the above experiments and discussions, we pool the *Conv*2 and *Conv*3 layers of temporal net. In this section, we compare the performance of two short-term pooling strategies and their combinations with long-term pooling. The results are summarized in Table 3. Different from trajectory pooling based on trajectories, line pooling directly pools stacked features from the convolutional feature maps along the time line (Zhao et al., 2015). From the results, we can see that the trajectory pooling improve 3.17% performance compared with the line pooling, due to the trajectory priority. From the first and the third rows (the second and the fourth rows) in Table 3, it is interesting to see a trend that the short-term pooling methods with rank pooling bring a substantial increase over themselves (2.08% for line pooling and 0.84% for trajectory pooling). This validates the effectiveness of long-term pooling for capturing the video-level dynamics. Overall, our hierarchical pooling strategy achieves the best performance 84.17% mAP.

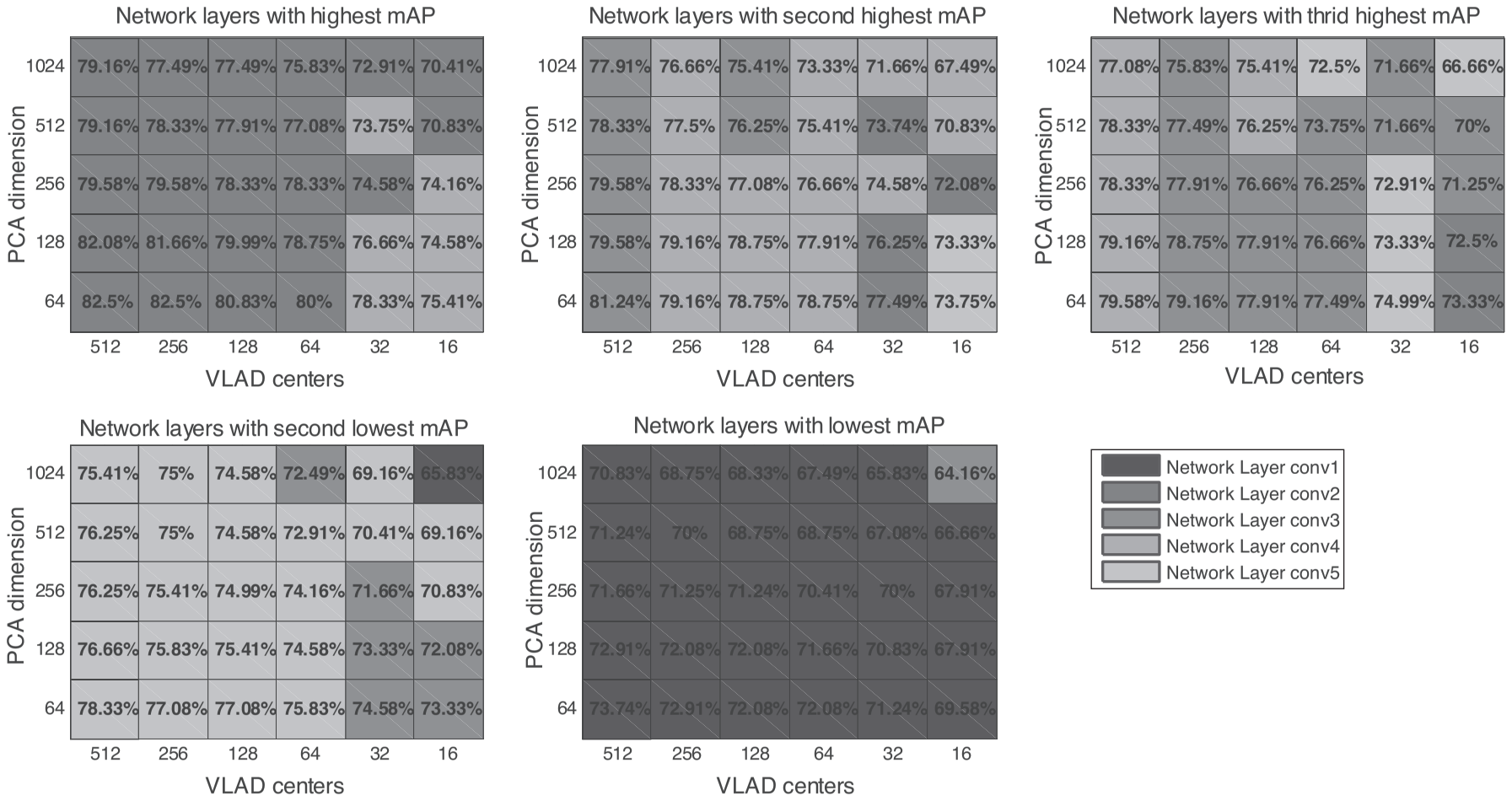

FIG. 10. Performance comparison of different single convolutional layers. In **(a–e)**, different colors stand for the different convolutional layers of ConvNets, while different rows and columns correspond to the different parameters of PCA dimension and VLAD centers, respectively. **(a)** Summarize the convolutional layers with the highest mAP scores for the parameter combinations. In the same way, **(b–e)** present the convolutional layers with the second highest mAP scores, the third highest mAP scores, the second lowest mAP scores, and the lowest mAP scores, separately.

Table 2. Performance Comparison of Convolutional Layer Combinations

Convolutional layer	*mAP (%)*
*Conv*2	82.5 ± 2.68
*Conv*3	81.24 ± 2.82
*Conv*4	79.58 ± 2.78
*Conv*2 + *Conv*3	**83.33 ± 2.65**
*Conv*3 + *Conv*4	80.41 ± 2.92
*Conv*2 + *Conv*4	81.25 ± 2.75
*Conv*2 + *Conv*3 + *Conv*4	82.08 ± 2.71

mAP, mean average precision.

Bold values are the best performance in the corresponding experiments.

5.5. Comparison with the existing methods

In this section, we evaluate the performance of our proposed framework in comparison with several existing algorithms. These algorithms are divided into three groups. The first one focuses on extracting the frame-level features for the cellular morphologies, containing shape parameters, Zernike moment, radial distance, and tree graph (Xiong and Iglesias, 2010; An et al., 2012; Alizadeh et al., 2016; Tsygankov et al., 2014). Then, we can obtain the cell dynamics for video clips by concatenating the variation (e.g., Euclidean distance) of frame-level features. The frame interval is specified as 20, while the Zernike moment up to 30.

The second group is to profile the short-term cell dynamics on several neighbor frames, such as shape context, optical flow, and our proposed trajectory-pooled deep-convolutional (TPDC) feature (Huang et al., 2013; Li et al., 2015b; Pang et al., 2015). In Huang et al. (2013), the short-term cell dynamics is represented by combining the radial distance and optical flow (RDOF) feature. Meanwhile, shape context and scale invariant feature transform flow are introduced in the TBoW framework (Pang et al., 2015). The subsequent temporal aggregation is also the straightforward concatenation or accumulation strategy. The rest methods concentrate on modeling the long-term cell dynamics based on the forementioned features. HMM for the long-term cell dynamics was reported in SAPHIRE (Stochastic Annotation of Phenotypic Individual-cell Responses) framework (Gordonov et al., 2016), while TBoW and local deformation pattern in our previous work (Pang et al., 2015; Li et al., 2015b). Our proposed framework is denoted as hierarchical-pooled deep-convolutional (HPDC) feature.

The experimental results (recognition precision, recall, and F-score measures) on test set are summarized in Table 4. In the first group, radial distance reaches higher performance than shape parameters and Zernike moment, because radial distance may preserve more subtle information of the cellular morphology. Tree graph is a variant of radial distance designed for the variation of cell protrusions. It does not achieve as good performance as radial distance, since lymphocytes in our data set may not have the explicit protrusions. From Table 4, we can find out that the methods in the second group possess better performance than those in the first group. It might be because these methods focus on the short-term cell dynamics instead of cellular morphology. Our TPDC feature improves more than 17.27% precision, 17.48% recall, and 18.43% F-score over Radial distance and RDOF feature.

As the approaches in the third group further model the long-term cell dynamics, most of them obtain better performance, except SAPHIRE. The reason might be that SAPHIRE uses shape parameters as the

Table 3. Performance Comparison of Different Pooling Strategies

Pooling strategies	*mAP (%)*
Line pooling	79.16 ± 2.76
Trajectory pooling	83.33 ± 2.65
Line pooling+rank pooling	81.24 ± 2.71
Trajectory pooling+rank pooling	**84.17 ± 2.67**

Bold values are the best performance in the corresponding experiments.

Table 4. Performance Comparisons (Precision, Recall, and F-Score, in Percentage) with Several Mainstreaming Methods

Methods	*Precision (%)*	*Recall (%)*	*F-score (%)*
Shape parameters	38.34 ± 14.16	45.45 ± 11.35	29.91 ± 13.52
Zernike moment	55.95 ± 10.16	60.65 ± 12.75	54.46 ± 11.49
Radial distance	61.66 ± 14.25	63.95 ± 13.14	55.81 ± 13.88
Tree graph	57.15 ± 12.36	54.40 ± 11.80	53.46 ± 10.30
RDOF feature	66.37 ± 12.95	65.85 ± 17.24	64.63 ± 14.31
LDP	81.72 ± 7.60	80.45 ± 8.46	79.09 ± 8.05
SAPHIRE	58.24 ± 7.86	57.65 ± 6.86	56.76 ± 7.20
DC-SAPHIRE	81.45 ± **3.38**	80.75 ± 3.74	80.65 ± **3.87**
TBoW	80.95 ± 11.79	79.95 ± 17.52	79.29 ± 12.58
TPDC feature (ours)	83.64 ± 4.63	83.33 ± **2.65**	83.06 ± 5.06
HPDC feature (ours)	**84.08** ± 4.15	**84.17** ± 2.67	**83.92** ± 4.06

HPDC, hierarchical-pooled deep-convolutional; LDP, local deformation pattern; RDOF, radial distance and optical flow; TBoW, temporal bag of words; TPDC, trajectory-pooled deep-convolutional.

Bold values are the best performance in the corresponding experiments.

feature of short-term cell dynamics, which can only roughly characterize cell dynamics. Therefore, we introduce our deep-convolutional feature into SAPHIRE framework (denoted as DC-SAPHIRE). The fact that DC-SAPHIRE reaches the third highest performance in Table 4 also proves the significance of capturing the long-term cell dynamics. At last, our proposed framework achieves a better performance (84.08% precision, 84.17% recall rate, and 79.29% F-score) in comparison with other existing algorithms.

6. CONCLUSIONS

In this article, we have presented a novel framework to analyze temporal dynamics of cells. This framework can automatically learn the video-wide cell dynamics with significant biological meaning from raw live-cell videos, because it shares the merits of both deep-learning technique and hierarchical temporal pooling algorithm. On the one hand, we have transferred a pretrained optical flow-based temporal net to the task of cell dynamic analysis, which embeds the cell dynamics into biologically meaningful feature maps. On the other hand, we have proposed hierarchical temporal pooling strategy to facilitate capturing short-term cell dynamics and modeling the video-wide evolution of temporal dynamics. Experimental results demonstrate that the proposed framework effectively captures the long-term spatiotemporal dynamics from the raw live-cell videos and outperforms existing methods on the cell video database.

ACKNOWLEDGMENT

The research reported in this publication was supported in part by the National Natural Science Foundation of China (61271112).

AUTHOR DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Alizadeh, E., Lyons, S.M., Castle, J.M., et al. 2016. Measuring systematic changes in invasive cancer cell shape using zernike moments. *Integr. Biol.* 8, 1183–1193.

An, X., Liu, Z., Shi, Y., et al. 2012. Modeling dynamic cellular morphology in images, 340–347. In: Ayache, N., Delingette, H., Golland, P., and Mori, K. (eds). *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, Berlin-Heidelberg.

Chechik, G., and Koller, D. 2009. Timing of gene expression responses to environmental changes. *J. Comput. Biol.* 16, 279–290.

Chen, C.L., Mahjoubfar, A., Tai, L.C., et al. 2016. Deep learning in label-free cell classification. *Sci. Rep.* 6, 21471.

Dunkers, J.P., Lee, Y.J., and Chatterjee, K. 2012. Single cell viability measurements in 3D scaffolds using in situ label free imaging by optical coherence microscopy. *Biomaterials* 33, 2119–2126.

Fan, R.E., Chang, K.W., Hsieh, C.J., et al. 2008. Liblinear: A library for large linear classification. *J. Mach. Learn Res.* 9, 1871–1874.

Feichtenhofer, C., Pinz, A., and Zisserman, A. 2016. Convolutional two-stream network fusion for video action recognition. *arXiv Preprint arXiv:1604.06573*.

Fernando, B., Gavves, E., Oramas, J., et al. 2017. Rank pooling for action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 773–787.

Gordonov, S., Hwang, M.K., Wells, A., et al. 2016. Time series modeling of live-cell shape dynamics for image-based phenotypic profiling. *Integr. Biol.* 8, 73–90.

Hasan, M., Choi, J., Neumann, J., et al. 2016. Learning temporal regularity in video sequences. *arXiv Preprint arXiv:1604.04574*.

Held, M., Schmitz, M.H., Fischer, B., et al. 2010. Cellcognition: Time-resolved phenotype annotation in high-throughput live cell imaging. *Nat. Methods.* 7, 747–754.

Hu, Y., and Yang, G. 2016. Sequence evolution under constraints: Lessons learned from sudoku. *J. Comput. Biol.* 23, 830–840.

Huang, Y., Liu, Z., Shi, Y., et al. 2013. Quantitative analysis of lymphocytes morphology and motion in intravital microscopic images, 3686–3689. In: IEEE (ed). *35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, New Jersey.

Jegou, H., Perronnin, F., Douze, M., et al. 2012. Aggregating local image descriptors into compact codes. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 1704–1716.

Johnson, G.R., Buck, T.E., Sullivan, D.P., et al. 2015. Joint modeling of cell and nuclear shape variation. *Mol. Biol. Cell.* 26, 4046–4056.

Kaviani, R., Merat, P., Moldovan, F., et al. 2016. An automated cell viability quantification method for low-resolution confocal images of closely packed cells based on a modified gradient flow tracking algorithm. *J. Microsc.* 261, 217–226.

Kotyk, T., Dey, N., Ashour, A.S., et al. 2015. Detection of dead stained microscopic cells based on color intensity and contrast, 57–68. In: Gaber, T., Hassanien, A.E., El-Bendary, N., and Dey, N. (eds). *The 1st International Conference on Advanced Intelligent System and Informatics (AISI)*. Springer, Berlin-Heidelberg.

Krizhevsky, A., Sutskever, I., and Hinton, G.E. 2012. Imagenet classification with deep convolutional neural networks, 1097–1105. In: Pereira, F., Burges, C.J.C., Bottou, L., and Weinberger, K.Q. (eds). *Advances in Neural Information Processing Systems (NIPS)*, Morgan Kaufman, Massachusetts.

Li, B., and Acton, S.T. 2007. Active contour external force using vector field convolution for image segmentation. *IEEE Trans. Image Process.* 16, 2096–2106.

Li, D., Shao, L., Chen, B.C., et al. 2015a. Extended-resolution structured illumination imaging of endocytic and cytoskeletal dynamics. *Science* 349, aab3500.

Li, H., Liu, Z., An, X., et al. 2014. Multi-classification of cell deformation based on object alignment and run length statistic, 3378–3381. In: IEEE (ed). *36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, New Jersey.

Li, H., Liu, Z., Pang, F., et al. 2015b. Analyzing dynamic cellular morphology in time-lapsed images enabled by cellular deformation pattern recognition, 7478–7481. In: IEEE (ed). *37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, New Jersey.

Liu, T.Y. 2009. Learning to rank for information retrieval. *Foundations Trends Inf Retrieval.* 3, 225–331.

Myers, C., and Rabiner, L. 1981. A level building dynamic time warping algorithm for connected word recognition. *IEEE Trans. Acoust. Speech Signal Process.* 29, 284–297.

Niederberger, T., Failmezger, H., Uskat, D., et al. 2015. Factor graph analysis of live cell–imaging data reveals mechanisms of cell fate decisions. *Bioinformatics*. btv040.

Obayemi, J., Danyuo, Y., Dozie-Nwachukwu, S., et al. 2016. Plga-based microparticles loaded with bacterial-synthesized prodigiosin for anticancer drug release: Effects of particle size on drug release kinetics and cell viability. *Mater. Sci. Eng. C* 66, 51–65.

Pang, F., Liu, Z., Li, H., et al. 2015. The measurement of cell viability based on temporal bag of words for image sequences, 4185–4189. In: IEEE (ed). *IEEE International Conference on Image Processing (ICIP)*. IEEE, New Jersey.

Rohde, G.K., Ribeiro, A.J., Dahl, K.N., et al. 2008. Deformation-based nuclear morphometry: Capturing nuclear shape variation in HeLa cells. *Cytometry A* 73, 341–350.

Sadanandan, S.K., Ranefall, P., and Wählby, C. 2016. Feature augmented deep neural networks for segmentation of cells, 231–243. In: Hua, G., and Jégou, H. (eds). *European Conference on Computer Vision (ECCV)*. Springer, Berlin-Heidelberg.

Shi, J., and Tomasi, C. 1994. Good features to track, 593–600. In: IEEE (ed). *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New Jersey.

Simonyan, K., and Zisserman, A. 2014. Two-stream convolutional networks for action recognition in videos, 568–576. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., and Weinberger, K.Q. (eds). *Advances in Neural Information Processing Systems (NIPS)*, Morgan Kaufman, Massachusetts.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. arXiv preprint; arXiv:1409.1556.

Smola, A., and Vapnik, V. 1997. Support vector regression machines, 155–161. In: Mozer, M.C., Jordan, M.I., and Petsche, T. (eds). *Advances in Neural Information Processing Systems (NIPS), Volume 9*..

Spiller, D.G., Wood, C.D., Rand, D.A., et al. 2010. Measurement of single-cell dynamics. *Nature* 465, 736–745.

Szegedy, C., Liu, W., Jia, Y., et al. 2015. Going deeper with convolutions, 1–9. In: IEEE (ed). *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New Jersey.

Tsygankov, D., Bilancia, C.G., Vitriol, E.A., et al. 2014. Cellgeo: A computational platform for the analysis of shape changes in cells with complex geometries. *J. Cell Biol.* 204, 443–460.

Wang, H., Kläser, A., Schmid, C., et al. 2013. Dense trajectories and motion boundary descriptors for action recognition. *Int. J. Comput. Vision.* 103, 60–79.

Wang, H., Oneata, D., Verbeek, J., et al. 2015a. A robust and efficient video representation for action recognition. *Int. J. Comput. Vision*. 119, 1–20.

Wang, K., Sun, W., Richie, C.T., et al. 2015b. Direct wavefront sensing for high-resolution in vivo imaging in scattering tissue. *Nat. Commun.* 6. Article #: 7276.

Wang, L., Qiao, Y., and Tang, X. 2015c. Action recognition with trajectory-pooled deep-convolutional descriptors, 4305–4314. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, Y., and Cong, J. 2016. An improved model of nonuniform coleochaete cell division. *J. Comput. Biol.* 23, 693–709.

Xiong, Y., and Iglesias, P.A. 2010. Tools for analyzing cell shape changes during chemotaxis. *Integr. Biol.* 2, 561–567.

Yuan, L., Zheng, Y.F., Zhu, J., et al. 2012. Object tracking with particle filtering in fluorescence microscopy images: Application to the motion of neurofilaments in axons. *IEEE Trans. Med. Imaging.* 31, 117–130.

Zach, C., Pock, T., and Bischof, H. 2007. A duality based approach for realtime tv-l 1 optical flow, 214–223. *In Joint Pattern Recognition Symposium*. Springer.

Zhao, S., Liu, Y., Han, Y., et al. 2015. Pooling the convolutional layers in deep ConvNets for action recognition. *arXiv Preprint arXiv:1511.02126*.

Zhong, Q., Busetto, A.G., Fededa, J.P., et al. 2012. Unsupervised modeling of cell morphology dynamics for time-lapse microscopy. *Nat. Methods* 9, 711–713.

Zhong, Q., Rüschoff, J.H., Guo, T., et al. 2016. Image-based computational quantification and visualization of genetic alterations and tumour heterogeneity. *Sci. Rep.* 6. Article #: 24146.

Address correspondence to:
Prof. Zhiwen Liu
Department of Information and Electronics
Beijing Institute of Technology
Beijing 100081
China

E-mail: zwliu@bit.edu.cn