Egocentric Temporal Action Proposals

Shao Huang, *Member, IEEE,* Weiqiang Wang, *Member, IEEE,* Shengfeng He, *Member, IEEE,* and Rynson W.H. Lau, *Senior Member, IEEE*

***Abstract*—We present an approach to localize generic actions in egocentric videos, called *temporal action proposals (TAPs)*, for accelerating the action recognition step. An egocentric TAP refers to a sequence of frames that may contain a generic action performed by the wearer of a head-mounted camera, e.g., taking a knife, spreading jam, pouring milk, or cutting carrots. Inspired by object proposals, this work aims at generating a small number of TAPs, thereby replacing the popular sliding window strategy, for localizing all action events in the input video. To this end, we first propose to temporally segment the input video into action atoms, which are the smallest units that may contain an action. We then apply a hierarchical clustering algorithm with several egocentric cues to generate TAPs. Finally, we propose two actionness networks to score the likelihood of each TAP containing an action. The top ranked candidates are returned as output TAPs. Experimental results show that the proposed TAP detection framework performs significantly better than relevant approaches for egocentric action detection.**

***Index Terms*—temporal action proposals, egocentric video processing, actionness estimation, temporal actionness network.**

I. Introduction

Wearable cameras, such as GoPro and Google Glass, enable users to record videos from a first-person perspective and are now widely used in various applications such as healthcare and education. These video data are typically referred to as *egocentric videos*. Their popularity leads to new research problems, e.g., gaze prediction [1], action recognition/detection [2], snap point prediction [3], social interaction understanding [4], photographer identification [5], future localization [6], environment functional attribute understanding [7], and visual recognition in autonomous driving [8]. Among these problems, temporal action detection is arguably one of the most important tasks in egocentric video analysis.

Automatically analyzing and localizing actions performed by the camera wearer from a long egocentric video are in great demand. To locate action events, existing works typically examine each frame and each possible clip length in a sliding window manner [9], [10], therein producing a large number of candidate clips of different lengths. These clips are then fed to the action classifier [11], [12] for action recognition. The main drawback of this sampling strategy is the large number of candidate clip lengths produced, e.g., more than 10,000 clip lengths for a 10 minute video, but only a small number of such clips may significantly overlap with the ground truth action events, causing a high computational overhead. This problem is analogous to the traditional object detection problem, which was an important research topic for decades [13], [14].

Motivated by the popular object proposal detection approach [15], [16], we propose a new concept called *temporal action proposals (TAPs)* to detect action events with a small number of proposals as a replacement for the widely used sliding window strategy for action recognition [11], [12]. Our proposal leads to a significant improvement in computational efficiency and detection accuracy. Inspired by the idea of merging superpixels to generate object proposals [15], [17], we first propose to segment the input egocentric video into action atoms, which are regarded as basic units of TAPs, based on the velocity/acceleration computed from the optical flow maps. We then propose various egocentric cues (hand position, eye gaze, motion blur and TAP length) and a hierarchical clustering algorithm to merge action atoms into candidate TAPs, aiming at covering all action events in the egocentric video. Finally, we train two networks to estimate how likely a candidate TAP is to contain a generic action event. To this end, we explore the spatial and temporal actionness networks for TAP scoring. The experimental results show that high detection rates (95% at IoU threshold$=0.85$) can be achieved with only a few TAPs (approximately 300 versus 10,000+ by sliding windows). Fig. 1 shows the overall pipeline of the proposed framework.

The main contributions of this work include the following:

- We propose a method to generate TAPs using two algorithms: an algorithm to segment the video into action atoms and an algorithm to hierarchically merge these atoms to form candidate TAPs with egocentric cues. High detection rates are achieved with a small number of TAPs.
- We present two networks for actionness estimation of TAPs, one based on spatial information from selecting representative frames and the other based on temporal information from constructing a scanline structure from each TAP. The results show that the proposed networks can effectively identify TAPs that contain action events.

II. Related Work

As this work is mainly motivated by object proposals, we first briefly summarize the latest works on object proposals. We then discuss related works on egocentric video analysis, action recognition and temporal action localization.

The work was supported in part by the National Key Research and Development Program of China (No. 2017YFB1002203), by the National Natural Science Foundation of China (No. 61772495, 61232013 and 61702194), by Beijing Advanced Innovation Center for Imaging Technology (No. BAICIT-2016009), and by two SRG grants from City University of Hong Kong (No. 7004676 and 7004889). (Corresponding author: Shengfeng He.)

Shao Huang is with the University of Chinese Academy of Sciences and City University of Hong Kong. E-mail: shaohuang6-c@my.cityu.edu.hk.

Weiqiang Wang is with the University of Chinese Academy of Sciences, China. Email: wqwang@ucas.ac.cn.

Shengfeng He is with the School of Computer Science and Engineering, South China University of Technology, China. E-mail: hesfe@scut.edu.cn.

Rynson W.H. Lau is with the Department of Computer Science, City University of Hong Kong, Hong Kong. E-mail: Rynson.Lau@cityu.edu.hk.

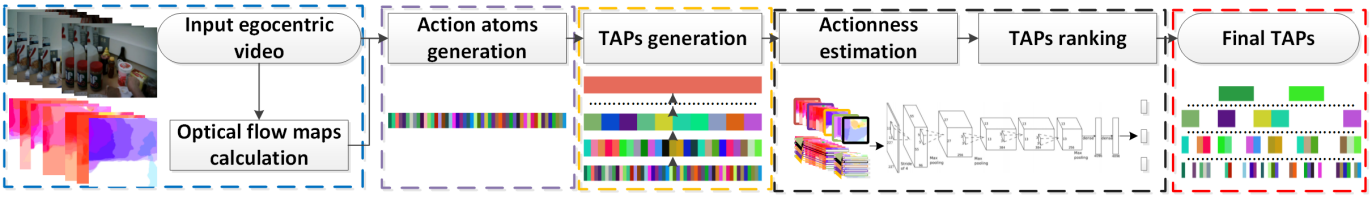


Fig. 1. The proposed framework. Given an egocentric video, we first compute optical flow maps for all frames, which are used to break down the video into action atoms. We then apply a hierarchical clustering algorithm to merge the action atoms to produce *temporal action proposals (TAPs)*. Finally, we propose two actionness networks to estimate the actionness of each TAP. The top ranked TAPs are outputted as the final proposals.

A. Image and Video Object Proposals

The goal of generating object proposals is to detect a small number of bounding boxes that cover all objects in an image. This approach replaces the time-consuming sliding window strategy, which was commonly used to produce image regions exhaustively for object recognition. As a result, more sophisticated features can be used in the recognition step. Many object proposal methods have been proposed recently [15], [16], [18], [19]. These methods either propose a scoring function to rank the candidate bounding boxes, such as in BING [16], or group superpixels with different criteria into larger regions as proposals such as in Selective Search [15]. A comprehensive survey can be found in [20].

The idea of generating object proposals is not limited to images. An intuitive extension is to detect object proposals from hand-held videos [21], where segments or bounding boxes are generated to cover all possible objects in each frame. In [22], action proposals are introduced. Each action proposal is a temporal series of spatial bounding boxes, i.e., a spatio-temporal video tube, which likely contains some human actions. Hence, they are a variant of video object proposals that skips static objects. Although we share a somewhat similar name with [22], we actually address a different problem. While they attempt to track each spatial object proposal across frames, we attempt to identify consecutive frames that may contain a generic action performed by the camera wearer. Due to the long recording time of an egocentric video, temporal localization is substantially more important than spatial localization. Hence, the proposed method differs from previous object proposal methods in terms of input video type (i.e., egocentric videos), output (i.e., TAPs), and the features used for analysis (i.e., egocentric cues).

B. Egocentric Video Analysis

Egocentric videos are different from hand-held videos in many ways, as they are typically captured by head-mounted cameras. As the camera wearer moves the head, the video content captured changes, indirectly reflecting the wearer's actions. This distinct property leads to many interesting research problems on egocentric videos. Li *et al.* [23] leverage implicit egocentric cues, such as hand/head position and motion, for gaze prediction. The detected gaze, together with egocentric cues, can be used for action recognition [2], [24]. Zhou *et al.* [25] propose an EM-like framework with a CNN for the pixel-level hand segmentation of egocentric videos. Yonetani *et al.* [4] propose the concept of micro-actions and micro-reactions for understanding the dynamics of social interactions with paired egocentric videos captured by the users during the interactions. Hoshen *et al.* [5] propose *photographer identification* to recognize the camera wearer from a few seconds of a video captured while walking. Park *et al.* [6] predict plausible future trajectories of ego-motion in egocentric stereo images. Rhinehart *et al.* [7] propose the concept of action maps for understanding the functional attributes of an environment by observing human interactions and the visual context of their actions in the environment. Xiong *et al.* [3] predict *snap points* in egocentric videos, which are frames that are likely intentionally captured by the camera wearer.

All the above works aim to analyze the egocentric videos to extract useful information for various applications. Our work shares a similar objective as some of these works. While it helps accelerate the action recognition task, the output TAPs can also be considered as a summary of activities in a long egocentric video.

C. Action Recognition

Action recognition is a fundamental problem in video understanding [26]. To leverage the information from both spatial and temporal domains, the motion trajectories of feature points, which can be obtained from dense optical flow maps [27], are commonly used for recognition [25].

To localize generic actions, the sliding window strategy is widely used in both spatial and temporal domains as a preprocessing step for recognition [11], [12]. However, as it is an exhaustive approach, it suffers from a high processing overhead. In this work, we address this problem by proposing a method to generate temporal action proposals (TAPs), i.e., consecutive frames that may contain a generic action performed by the camera wearer, for egocentric videos. This idea could be extended to general videos, with different cues.

The works most similar to ours are [26], [28], [29], [30], which benefit from deep convolutional neural networks and utilize optical flow for action recognition. Feichtenhofer *et al.* [26] study ways to fuse ConvNet towers spatially and temporally for action recognition in hand-held videos. Ma *et al.* [28] propose a twin stream network architecture for action recognition in egocentric videos, where one stream analyzes appearance information and the other stream analyzes motion information. Singh *et al.* [29] propose a network that considers egocentric cues, including hand pose, head motion and a saliency map, for end-to-end learning and classification of the wearer's actions. Weinzaepfel *et al.* [30] propose an approach to detect proposals at the frame level and score them with a combination of static and motion CNN features,

followed by a tracking-by-detection algorithm for temporal action localization. Our work differs significantly from these works in the following aspects:

1) The objective and output are different – TAPs with high probabilities to contain generic actions.
2) The framework is different – TAP generation followed by actionness estimation.
3) The network architecture is different – two networks are investigated, including temporal network that has not been studied before.
4) Egocentric cues are formulated in a different way.

D. Temporal Action Localization

Temporal action localization aims at finding the start and end of an action in a hand-held untrimmed video [31], [32]. Shou *et al.* [32] propose three segment-based 3D ConvNets to address temporal action localization. Yuan *et al.* [33] propose a Pyramid of Score Distribution Feature (PSDF) and explore the inter-frame consistency by incorporating PSDF into RNNs to handle the uncertainty of action occurrence and utilize the information from different scales. Gaidon *et al.* [31] propose a model based on a sequence of atomic action units to localize actions. Oneata *et al.* [34] apply the approximations of Fisher vector (FV) normalization, which enables the use of branch-and-bound search for temporal action localization. However, hand-held videos present very different properties from egocentric videos. Thus, these methods cannot handle egocentric motions. Moreover, our objective is to generate a small number of ranked proposals, instead of classification/localization.

III. Temporal Action Proposal Generation

In this section, we first present a method to segment an input video into action atoms (Section III-A) and then propose a hierarchical clustering method to merge action atoms into TAPs by jointly considering different egocentric cues (Section III-B).

A. Action Atoms

Action atoms are the basic units of a video in the temporal domain, analogous to the superpixels of an image in the spatial domain. Each action atom contains a cluster of frames that depict a tiny period of relatively minor/fine motion. A TAP is then constructed by merging similar action atoms together. The objective of having action atoms is to improve the efficiency of subsequent steps.

To this end, we consider the amount of motion between each pair of successive frames, as motion is an important indicator of an action. We compute the optical flows across frames using the large displacement optical flow method [35]. The total amount of optical flow is used as an indicator of the amount of motion to determine if we should insert a break into a motion atom. Note that all frames of the video are divided into action atoms. We do not drop any frames here, and there are no overlaps among the atoms. Because egocentric videos contain more motion patterns than hand-held videos, such as head motion, body motion, and hand motion, many action atoms may be produced, some with only a few frames. However, we do not expect this to be a problem here due to our hierarchical clustering algorithm in the next step. Nevertheless, we have performed an experiment to compare two approaches for detecting action atom breaks in Section VI-A. One approach is based on computing velocity changes, and the other approach is based on computing acceleration changes. The experiment attempts to investigate if these two different approaches may affect the accuracy of the detected TAPs.

Input: A video sequence containing M frames and thresholds α and β;
Output: The set of action atoms $\mathbf{A_c}$;
Initialization: $\mathbf{A_c} = \varnothing$; timestamp pointer $t = 1$, index pointer $k = 1$, accumulation variable $\xi = 0$; atom $\mathbf{a}_k = [1 \quad 1]$ contains two variables - the indices of the first and last frames;
for $t \leq M$ **do**
  Divide the tth optical flow map into 4×4 regular grids;
  Compute the median magnitude for each grid;
  Sum all grids as the magnitude ν_t for the tth frame;
  Update the accumulation variable $\xi \leftarrow \xi + \nu_t$;
  if $\nu_t \geq \alpha$ *or* $\xi \geq \beta$ **then**
    Update the last frame index of $\mathbf{a}_k$ as t;
    Add $\mathbf{a}_k$ into the set $\mathbf{A_c}$;
    Initialize the next atom $\mathbf{a}_{k+1} = [t+1 \quad t+1]$;
    Update $\xi = 0, \ k \leftarrow k + 1$;
  end
  Update $t \leftarrow t + 1$;
end

Algorithm 1: Pseudo-code for generating action atoms.

The velocity-based approach directly utilizes motion information to separate action atoms, i.e., grouping consecutive frames with small velocity values as an action atom. This is based on the assumption that the wearer's head tends to remain relatively stationary when performing an action. The acceleration-based approach, on the other hand, utilizes velocity changes (acceleration) to separate action atoms, i.e., grouping consecutive frames with small velocity changes as an action atom. This is based on the assumption that the change in the wearer's movement velocity may signal a potential change in action. Given a frame F_t at timestamp t, we first convert the computed optical flows into the velocity/acceleration magnitude of each pixel, i.e., $\sqrt{(v_x^2 + v_y^2)}$ for velocity/acceleration, where v_x and v_y are the velocity/acceleration values in the x and y axes. F_t is then divided into regular grids, and each grid is represented by the median velocity/acceleration magnitude of all pixels within the grid. All the median velocity/acceleration magnitudes of F_t are averaged to produce the overall value ν_t for the frame.

The set of action atoms $\mathbf{A_c}$ are constructed in a frame-wise manner. F_t belongs to an action atom if ν_t is smaller than α and the sum of all νs of the action atom is smaller than β. As vigorous movements may appear at times that typically denote the start of a new action, α is a frame-variation threshold for segmenting these abrupt motions. The atom-variation threshold β is used to limit the maximum length of each atom, not to bound piecewise-like motions. The values of these parameters are the same across all video frames, and the constraint $\alpha = \frac{1}{2}\beta$ is set in our experiments. In this work, each optical flow map is empirically divided into

Input: The set of action atoms $\mathbf{A_c}$;
Output: The set of TAPs $\mathbf{A_p}$;
Initialize $\mathbf{A_p} = \mathbf{A_c}$;
Initialize the score set $\Theta = \varnothing$;
for *each pair of successive TAPs* ($\mathbf{a_{p_i}}$,$\mathbf{a_{p_j}}$) **do**
  Compute the score $\theta(\mathbf{a_{p_i}}, \mathbf{a_{p_j}})$;
  Add the score $\theta(\mathbf{a_{p_i}}, \mathbf{a_{p_j}})$ to the score set Θ;
end
while $\Theta \neq \varnothing$ **do**
  Find the highest score $\theta(\mathbf{a_{p_i}}, \mathbf{a_{p_j}})$, i.e., $\max\{\Theta\}$;
  Merge the corresponding TAP pair, i.e., $\mathbf{a_{p_o}} = \mathbf{a_{p_i}} \cup \mathbf{a_{p_j}}$;
  Remove scores related to $\mathbf{a_{p_i}}$ and $\mathbf{a_{p_j}}$ from the set Θ;
  Compute scores between $\mathbf{a_{p_o}}$ and its two neighboring TAPs;
  Add the new scores to the set Θ;
  Add $\mathbf{a_{p_o}}$ to the set $\mathbf{A_p}$;
end

Algorithm 2: Pseudo-code for the hierarchical clustering algorithm.

4×4 regular grids. While fewer grids (e.g., 3×3) typically lead to inaccurate motion combinations, more grids (e.g., 5×5) may decrease the robustness to noise. Algorithm 1 shows the pseudo-code for generating action atoms.

We have conducted an experiment to compare the effects of the two approaches on TAP generation. Our results in Sec. VI-A show that the velocity-based approach outperforms the acceleration-based approach due to the typically irregular and non-uniform motions of the camera wearer.

B. Temporal Action Proposals

The objective of TAP generation is to obtain the candidates containing generic actions performed by the camera wearer. As discussed in Section III-A, each action should contain one or multiple action atoms. Hence, we propose a hierarchical clustering algorithm to generate TAPs by regarding the action atoms obtained in the previous step as the bottom-level TAPs. We compute a score for each pair of adjacent action atoms (or TAPs) based on four egocentric cues: hand position, eye gaze, motion blur and TAP length (i.e., number of frames). The pair with the highest score will be merged. The scores for the merged TAP with its neighboring TAPs are recomputed, and another pair with the highest score will be selected for merging. This merging process will continue until all TAPs of the video are merged into one. All TAPs produced together with the action atoms form the TAP hierarchy. Algorithm 2 shows the pseudo-code of the hierarchical clustering algorithm.

The remainder of this subsection discusses our four egocentric cues used in the hierarchical clustering algorithm.

1) Hand Position: Our hands play an important role in human actions, as we use them to interact with objects, e.g., taking a knife and opening a bottle. Recently, some impressive works have been done on egocentric hand detection [23], [24], [25], [36], [37]. In [36], the background is assumed to be static so that the optical flow patterns can be used for segmentation. However, most egocentric videos do not have static backgrounds, as the wearers tend to walk around during the capture such as those in the GTEA dataset [36]. Moreover, [23], [37] assume no social interactions in the videos to simplify the detection and thus perform poorly in scenes where the wearer interacts with others such as in the EgoHands dataset [24]. Although the deep learning approaches [24], [25] achieve promising results, their performances are significantly affected by the insufficient amount of training data such as in the GTEA Gaze dataset [38].

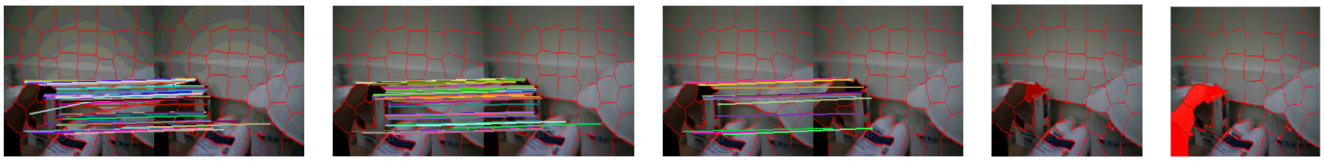
To address the performance and efficiency limitations of existing methods, we propose an efficient hand detection algorithm here by differentiating hand motion from the overall frame motion, which is training independent. We first use SLIC [39] to over-segment the input frame into superpixels. The ORB [40] descriptor, which is rotation invariant and resistant to noise, is then used for feature matching to reduce the computational cost. We then remove the obvious false matches, whose positions differ largely in two successive frames. The set of candidate matches is referred to as $\mathbf{M}$.

As suggested by [36], [37], the motion in egocentric videos is mainly affected by the mounted camera and hands. While all objects in the background are affected by head/camera motions, only hand regions are affected by hand motions. Hence, most matches in agreement with the head/camera motion distribute across the whole frame, while hand-related matches only distribute in a few regions. We calculate the homography matrix from the candidate matches with RANSAC. Camera-related matches $\mathbf{M}_c$ would then become the "inliers" (see Fig. 2(b)). Hence, the set of hand-related matches can be roughly estimated by $\mathbf{M}_h = \mathbf{M} - \mathbf{M}_c$ (see Fig. 2(c)).

This solution, however, may fail to extract all hand-related matches and can be noisy if the motions caused by the camera and hands are indistinguishable. Hence, it is problematic to directly regard the superpixels containing $\mathbf{m} \in \mathbf{M}_h$ as hand regions. Instead, we localize hand seed regions based on the distribution of $\mathbf{M}_h$. Specifically, we count the number of hand-related matches within each superpixel. Those with higher numbers than their neighboring superpixels are considered as seeds. Thus, two seeds cannot be adjacent to each other. We further filter the seeds by removing those with the number of hand-related matches smaller than 0.1μ, where μ is adaptively set to the average number of hand-related matches of all frames in the video. Typically, only one or two seeds are found in a frame with these steps (see Fig. 2(d)). (Note that if no seeds are found in a frame, we assume that it contains no hands, and the subsequent step for computing hand regions will be skipped.) We then estimate the score of each superpixel in the frame for being a hand region by considering the appearance and localization factors as follows.

Appearance: As suggested by popular datasets such as GTEA [36], EgoHands [24] and GTEA Gaze(+) [38], multiple hands could simultaneously appear in an egocentric video, but different parts of a hand share a similar appearance, e.g., the fingers and the back of a hand. Moreover, the appearance of a hand seldom changes abruptly in an egocentric video, as the illumination tends to be stable. With these observations, we compute spatial and temporal appearance scores based on appearance similarity using a normalized color histogram, $\mathbf{h} = [h_1, h_2, \cdots, h_P]$, with $P = 16$ bins for improved efficiency:

- *Spatial appearance score* $\mathrm{AP_S}(\mathbf{s_k})$ measures the appear-



(a) All matches using ORB (b) Camera-related matches $\mathbf{M}_c$ (c) Hand-related matches $\mathbf{M}_h$ (d) Seed (dark red) (e) Hand (red)

Fig. 2. Steps for localizing the hand regions. Color lines show the matches between successive frames.

ance similarity between each superpixel $\mathbf{s}_k$ and the most similar seed, both in the current frame, as

$$\mathrm{AP_S}(\mathbf{s_k}) = \exp(-\min_{\phi=1,\cdots,\omega} \sum_{j=1}^{P} (h_{k,j} \log \frac{h_{k,j}}{\hat{h}_{\phi,j}} + \hat{h}_{\phi,j} \log \frac{\hat{h}_{\phi,j}}{h_{k,j}})), \quad (1)$$

where $\mathbf{s}_k$ refers to the kth superpixel in the current frame, $h_{k,j}$ refers to the jth bin of the histogram on $\mathbf{s}_k$, and $\hat{h}_{\phi,j}$ refers to the jth bin of the histogram on the ϕth seed. ω is the number of seeds.

- *Temporal appearance score* $\mathrm{AP_T}(\mathbf{s_k})$ measures the similarity between each superpixel $\mathbf{s}_k$ in the current frame and the appearance models $\mathbf{A}^{t-1}$ updated from the previous frames as

$$\mathrm{AP_T}(\mathbf{s_k}) = \exp(-\min_{v=1,\cdots,\varphi} \sum_{j=1}^{P} (h_{k,j} \log \frac{h_{k,j}}{a_{v,j}^{t-1}} + a_{v,j}^{t-1} \log \frac{a_{v,j}^{t-1}}{h_{k,j}})), \quad (2)$$

where $a_{v,j}^{t-1}$ denotes the jth bin of the vth appearance model $\mathbf{a}_v^{t-1} \in \mathbf{A}^{t-1}$ at previous timestamp $t-1$, and φ denotes the number of appearance models. We formulate appearance update based on the following assumption: the contribution of a frame is related to its distance from the current timestamp in the temporal domain. Because multiple hands may appear in an egocentric video, the computed hand regions only contribute to their relevant models. Specifically, each region at timestamp $t-1$ is assigned to the most similar model based on KL divergence. The vth model $\mathbf{a}_v^{t-2} \in \mathbf{A}^{t-2}$ with n_v relevant components is updated as

$$\mathbf{a}_v^{t-1} = \delta \frac{1}{n_v} \sum_{i=1}^{n_v} \bar{\mathbf{h}}_{v,i}^{t-1} + (1-\delta)\mathbf{a}_v^{t-2}, \quad (3)$$

where $\delta = 0.4$ in this work, and $\bar{\mathbf{h}}_{v,i}^{t-1}$ is the ith relevant region assigned to $\mathbf{a}_v^{t-2}$ at timestamp $t-1$. For a model with no assigned relevant regions, we set $\mathbf{a}_v^{t-1} = \mathbf{a}_v^{t-2}$.

Location: A seed is surrounded by a relatively large number of hand-related matches in a local area, i.e., with a high chance of being located inside a hand. Thus, hand regions are expected to be close to at least one seed. Moreover, the hand location tends to change slowly between two successive frames due to the small frame interval. Based on these observations, we compute the spatial and temporal location scores from the perspective of position:

- *Spatial location score* $\mathrm{MM_S}(\mathbf{s_k})$ measures the Euclidean distance between each superpixel $\mathbf{s}_k$ and the nearest seed in the current frame to ensure proximity between the hand region and the seed as

$$\mathrm{MM_S}(\mathbf{s_k}) = \exp(-\min_{\phi=1,\cdots,\omega} \frac{\|\mathbf{c_k} - \hat{\mathbf{c}}_\phi\|_2}{\mathrm{d}}), \quad (4)$$

where $\mathbf{c}_k$ and $\hat{\mathbf{c}}_\phi$ are the centers of the superpixel $\mathbf{s}_k$ and the ϕth seed, respectively, in the current frame, and d is the diagonal length of the image.

- *Temporal location score* $\mathrm{MM_T}(\mathbf{s_k})$ measures the Euclidean distance between each superpixel $\mathbf{s}_k$ in the current frame and the nearest hand region in the previous frame as

$$\mathrm{MM_T}(\mathbf{s_k}) = \exp(-\min_{\varsigma=1,\cdots,\rho} \frac{\|\mathbf{c_k} - \bar{\mathbf{c}}'_\varsigma\|_2}{\mathrm{d}}), \quad (5)$$

where $\bar{\mathbf{c}}'_\varsigma$ is the center of the ςth hand region and ρ is the number of hand regions, both in the previous frame.

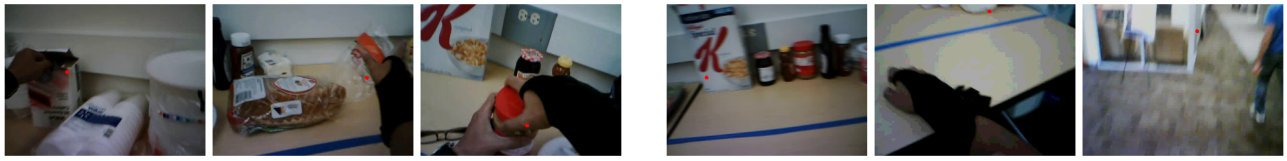
Hand Scoring: Each superpixel is assigned a likelihood value ι of belonging to the hand region by combining the above scores through the weighted summation as

$$\iota = 0.3[\mathrm{AP_S}(\mathbf{s_k}) + \mathrm{AP_T}(\mathbf{s_k}) + \mathrm{MM_T}(\mathbf{s_k})] + 0.1\mathrm{MM_S}(\mathbf{s_k}). \quad (6)$$

$\mathrm{MM_S}(\mathbf{s_k})$ is assigned with a smaller weight of 0.1, compared with other terms, as this term is less reliable in certain situations due to the widely distributed hand regions (e.g., the hand occupies a large area in the visual field when the wearer is eating or drinking). $\mathrm{AP_S}(\mathbf{s_k})$, $\mathrm{AP_T}(\mathbf{s_k})$ and $\mathrm{MM_T}(\mathbf{s_k})$ have very similar impacts on the performance in our experiments and are thus assigned the same weight. Starting from the seeds, we gradually merge neighboring superpixels with the highest likelihood values to form the hand regions (see Fig. 2(e)). This process terminates when the highest likelihood value is below a pre-defined threshold or when there is an obvious drop in the likelihood value indicating that the superpixel does not belong to the confirmed hand regions, e.g., 0.6 times lower than that of any hand region. The hand may occupy different areas in different frames, e.g., a hand is close to the mounted camera while eating but far from the camera while chopping. Hence, only hand regions participate in the computation of the likelihood of a frame h, therein using their mean likelihood values computed from Eq. 6 to handle the problem with different hand sizes. In particular, a frame containing no seeds is assigned a value of 0.

The score $\mathrm{S_{hand}}(\mathbf{a_{p_i}}, \mathbf{a_{p_j}})$ for the hand position between two adjacent TAPs, $\mathbf{a}_{\mathbf{p}_i}$ and $\mathbf{a}_{\mathbf{p}_j}$, is computed as

$$\mathrm{S_{hand}}(\mathbf{a_{p_i}}, \mathbf{a_{p_j}}) = \frac{\sum_{\mathrm{m} \in \text{frames in } \mathbf{a_{p_i}}} \mathrm{h_m} + \sum_{\mathrm{n} \in \text{frames in } \mathbf{a_{p_j}}} \mathrm{h_n}}{|\mathbf{a_{p_i}}| + |\mathbf{a_{p_j}}|}, \quad (7)$$



(a) Action frames (b) Non-action frames

Fig. 3. Examples of eye gaze points (red dots). Eye gaze localizations are often highly correlated with the hand location in the action frames.

where $|\mathbf{a}_{\mathbf{p}_i}|$ and $|\mathbf{a}_{\mathbf{p}_j}|$ denote the total numbers of frames in $\mathbf{a}_{\mathbf{p}_i}$ and $\mathbf{a}_{\mathbf{p}_j}$, respectively, and $\mathrm{h_m}$ and $\mathrm{h_n}$ denote the mean likelihood values for the mth and nth frames, respectively, in the two TAPs. A high $\mathrm{S_{hand}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j})$ indicates that most frames in the TAP pair likely contain the hand. Note that these computed hand regions also help estimate the eye gaze score.

2) Eye Gaze: The eye fixation of the camera wearer is another important cue for discovering ego-actions when exploring eye-object interactions. As suggested by the GTEA Gaze(+) dataset [38], eye gaze is typically correlated with hand position, as the gaze points often lie on the hand or the target object in an ego-action. The wearer is less likely to be performing an action otherwise. Fig. 3 shows some eye gaze examples of action and non-action frames. We follow the setting of [2], where the GT gaze is captured by a wearable eye tracker. Therefore, we integrate the eye gaze information if it is available in the dataset; otherwise, we simply ignore this cue. Score $\mathrm{S_{gaze}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j})$ for the eye gaze is defined as

$$\mathrm{S_{gaze}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j}) = \exp(-\frac{\kappa_i + \kappa_j}{2\mathrm{d}}), \tag{8}$$

where κ_i and κ_j are the mean distances between the gaze locations and the nearest hand regions in $\mathbf{a}_{\mathbf{p}_i}$ and $\mathbf{a}_{\mathbf{p}_j}$, respectively, and d is the diagonal length of the image.

3) Motion Blur: A previous work [3] suggests that a frame with significant motion blur is caused by unintentional camera movement, which often indicates that it contains no ego-actions, with some exceptions, such as chopping, that may cause the head to shake. Our observation is that the motion of the camera wearer tends to be relatively slow during an action due to concentration. Rapid motions, which cause blurriness, such as turning the head in a different direction, typically occur between two actions. We extract blurriness [41] as a descriptor to represent motion blur. The score $\mathrm{S_{blur}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j})$ for the motion blur between adjacent TAPs, $\mathbf{a}_{\mathbf{p}_i}$ and $\mathbf{a}_{\mathbf{p}_j}$, is defined as

$$\mathrm{S_{blur}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j}) = \frac{\sum_{\mathrm{m} \in \text{frames in } \mathbf{a}_{\mathbf{p}_i}} \mathrm{s_m} + \sum_{\mathrm{n} \in \text{frames in } \mathbf{a}_{\mathbf{p}_j}} \mathrm{s_n}}{|\mathbf{a}_{\mathbf{p}_i}| + |\mathbf{a}_{\mathbf{p}_j}|}, \tag{9}$$

where $\mathrm{s_m} = 1 - \mathrm{b_m}$, $\mathrm{s_n} = 1 - \mathrm{b_n}$, and $\mathrm{b_m}$ and $\mathrm{b_n}$ denote the estimated blurriness of frames m and n, respectively, and range between 0 and 1. The blurriness of a frame is computed based on the discrimination between different levels of blur perceived from the frame. It is referred to as the no-reference perceptual blur metric [41].

4) TAP Length: Similar to the problem in segmentation-based object proposals [15], a few TAPs with high scores may gobble up other neighboring TAPs to form increasingly larger TAPs, resulting in poor diversity. To address this problem, we encourage merging short TAPs first. Score $\mathrm{S_{len}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j})$ for the TAP length is defined as

$$\mathrm{S_{len}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j}) = \exp(-\frac{\mathrm{l_i} \times \mathrm{l_j}}{\mathrm{l_v}^2}), \tag{10}$$

where l_i and l_j are the lengths (or numbers of frames) of $\mathbf{a}_{\mathbf{p}_i}$ and $\mathbf{a}_{\mathbf{p}_j}$, respectively, and $l_{\mathbf{v}}$ is the total length of the video.

5) Combining All Cues: Finally, we compute a merging score for each pair of adjacent TAPs by considering a weighted sum of all the cues discussed above. The idea is that two successive TAPs have high priority to be merged first if more of the following conditions are satisfied: a hand appears within the frame, the gaze lies on the hand, the blurriness is low, and the TAPs are short. This merging score is thus computed as

$$\begin{aligned}\theta(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j}) = \tau \times [\mathrm{S_{hand}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j}) + \mathrm{S_{gaze}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j})] + \\ (0.5 - \tau) \times [\mathrm{S_{blur}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j}) + \mathrm{S_{len}}(\mathbf{a}_{\mathbf{p}_i}, \mathbf{a}_{\mathbf{p}_j})].\end{aligned} \tag{11}$$

We set $\tau = 0.4$ to prefer hand position and eye gaze, as they reveal distinctive characteristics for egocentric videos. On the other hand, motion blur depicts the degree of motion, and the TAP length enhances the diversity. Both of these metrics are generally useful for hand-held videos although not specifically for egocentric videos. TAPs with the highest score are merged in each iteration, as shown in Algorithm 2. As shown in Section V-A, overlapping actions are rare in the GTEA Gaze dataset. Thus, a TAP is assumed to be contained by no more than one action for simplification. Hence, scores related to the merged TAP pair are all removed in our experiments to ensure that the small number of generated TAPs are diversified. In other scenarios/datasets where overlapping actions may be common, we can remove the score of the merged TAP pair only and retain other related scores to ensure that an action can be contained by multiple TAPs.

IV. Actionness Estimation

Actionness represents how likely a TAP is to contain an action, and estimating actionness helps reduce the number of TAPs that must be fed to the classifier. Note that although we share a similar name "actionness" with [42], [43], the definitions are completely different: [42] defines "actionness" as intentional bodily movement of biological agents and a subclass of general motion, while [43] proposes an "actionness map" to localize actions in a single frame by estimating how likely a region is to be involved in an action. In this section, we explore different network architectures for actionness estimation. The design principle of our networks is to extract a fixed amount of distinct information to represent

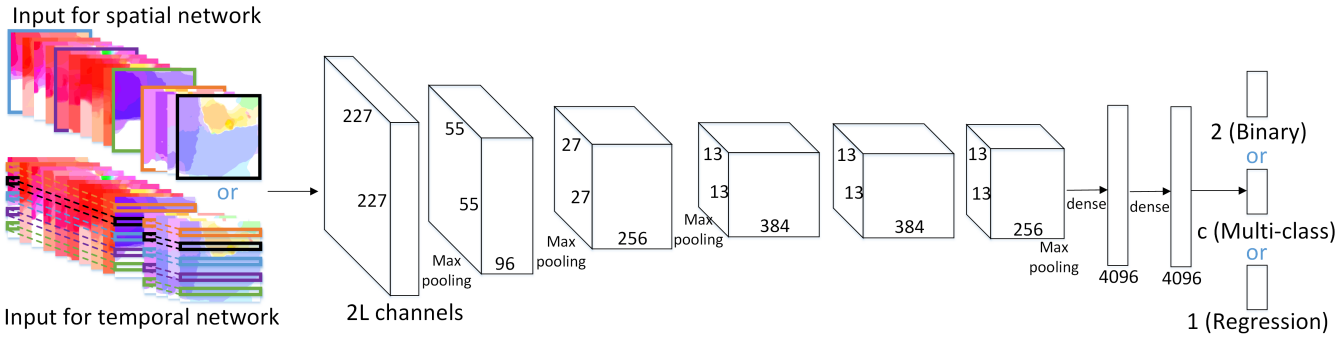


Fig. 4. The architecture of our spatial or temporal actionness network. Only one of the two inputs and only one of the three outputs are used in a network. The "multi-class" output has c categories, including the additional non-action category in the dataset. Different combinations of the inputs and outputs produce six different networks for actionness estimation. All these networks are based on AlexNet [13].

the input TAP for actioness estimation. To this end, we propose to selectively extract spatial or temporal information of a TAP using a CNN, which is initialized with AlexNet [13]. As the optical flow maps [35] contain motion information, we use them as input for actionness estimation. Note that a frame has two flow maps, i.e., for the motions in the x and y axes, separately.

A. Spatial Actionness Network

Given an input TAP, the most intuitive way to evaluate it is to feed all flow maps to the CNN. However, each TAP contains a different number of maps ranging from tens to thousands, which prevents us from feeding the entire TAP to the CNN, as the input layer has a fixed number of channels for all TAPs. Hence, we propose to select a fixed number of flow maps to represent each TAP. Because the first and last maps of a TAP depict the boundaries of a potential generic action, we always select them. We then select the other maps in a motion clustering manner:

1) We divide each optical flow map into a 4×4 regular grid and compute the median motion of each cell to form a 16×2 matrix, i.e., 16 cells of x and y motions.
2) We then segment the TAP into $L-2$ partitions based on the variances of these matrices.
3) Finally, we select the temporally centered map from each partition, combined with the first and last maps of the TAP, to produce a total of $2L$ maps.

These maps are stacked in temporal order to construct the input data for our spatial actionness network with $2L$ channels. In this work, L is set to 10 for two reasons. First, some annotated actions are extremely short, e.g., $10 \sim 20$ frames, and problems may arise if L is set to a value larger than the number of frames in a TAP, e.g., virtual flow maps will need to be generated through averaging the adjacent flows. However, such an idea fails to generate the "true and representative map", which is expected to represent the transient motion. Second, the performance will not be improved significantly in our experiments when a larger value of L is set, but the efficiency will be strongly affected as the number of input channels is increased, which would cause more weights in the network, resulting in longer training and testing times.

The main advantage of this TAP representation is that the spatial information of the selected maps is complete so that important information, such as the hands and target objects, is included in the actionness estimation. The downside is that most of the temporal information is discarded, as the selected flow maps are discontinuous.

B. Temporal Actionness Network

If we consider an optical flow sequence as a 3D volume with x, y and t dimensions, our spatial actionness network selects flow maps along the t dimension to form channels. To preserve the temporal information, our temporal actionness network selects scanlines to form channels, and scanlines are selected along the y dimension. In this way, the network receives continuous optical flow information:

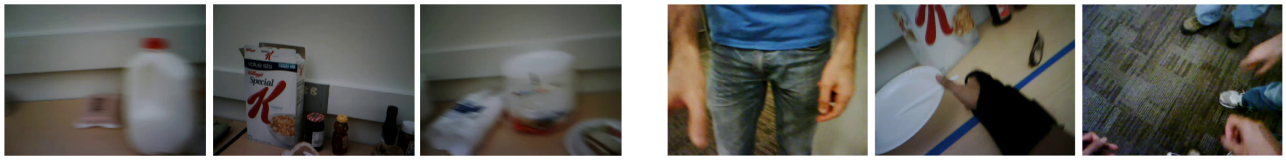
1) Given a TAP, we uniformly select L scanlines from each flow map of the TAP, i.e., the locations of extracted scanlines are the same for all maps, and the width of each scanline is the same as the width of each map.
2) We combine the selected scanlines from the same y location in all frames to form a scanline structure.
3) We resize this structure to a 227×227 matrix to form an input channel, as each TAP may have a different length.

This results in a total of $2L$ channels (2 referring to the x and y motions) as input to the temporal actionness network. Note that for the spatial network, the input is constructed by selecting *representative* maps. For the temporal network, we do not assume that some scanlines are more *representative* than others, and we select scanlines from all frames uniformly. The main advantage of this representation is that the optical flow information is continuous, and thus, the motions across the TAP are retained; however, the disadvantage is that we may lose some horizontal motions of very small objects.

C. Network Output

The above spatial and temporal actionness networks can be used for various tasks by setting different output layers. We explore three architectures for three different tasks.

- **Binary classification network**: Actionness estimation is considered as a binary classification problem, i.e., the



(a) Action frames without hands.

(b) Non-action frames with hands.

Fig. 5. Example frames with hands. Hand appearance is not reliable in determining whether a frame contains an action.

TABLE I
STATISTICS OF DIFFERENT ACTION CATEGORIES IN THE TRAINING AND TESTING VIDEOS. THE FIRST TWO ROWS SHOW THE FREQUENCY OF GT ACTIONS. THE LAST TWO ROWS SHOW THE FREQUENCY OF THE GENERATED POSITIVE SAMPLES.

Action category	Take	Open	Scoop	Spread	Close	Put	Sandwich	Pour	Fold	Press	Cut	Clean
Training GT	160	40	21	29	31	43	5	21	1	5	2	1
Testing GT	34	9	11	12	9	11	5	4	0	0	0	0
Positive training samples	1,446	332	188	498	227	202	45	227	6	30	37	12
Positive testing samples	361	79	134	204	85	43	40	54	0	0	0	0

input TAP containing a generic action or not. Actions such as taking, folding and pressing are considered as *generic actions*, and the network aims to learn the general motion patterns of these actions. Unlike 2D object proposal detection, in which the IoU threshold is typically set to 0.5 as it has 2 degrees-of-freedom, our proposal detection problem has only 1 degree-of-freedom, and the IoU threshold between a valid TAP and the GT action in the temporal domain should be higher. In this work, we set the IoU threshold to 0.85 for accuracy purposes.

- **Multi-class classification network**: Instead of estimating whether a TAP contains a generic action, this network directly predicts the category of action performed in the input TAP by assuming each specific type has distinctive motion patterns. As such, this network learns the motion features from the same category. We would like to see if action recognition can be directly addressed. The IoU threshold for positive samples is also set to 0.85.
- **Regression network**: Similar to the binary classification network, a TAP is considered as an action or non-action TAP. However, the difference is that this network is trained as a regression problem with more specific guidance, i.e., the IoU between a TAP and the nearest GT action is used as the label.

Fig. 4 shows the network architecture with different inputs and outputs. The effectiveness of these networks is investigated in Section VI-D.

V. IMPLEMENTATION DETAILS

In this section, we present some important implementation details of our networks. We first discuss the dataset (Section V-A) and how we prepare the training samples (Section V-B). We then summarize the training and testing details (Section V-C).

A. Dataset

We use the GTEA Gaze dataset [38] for training and testing. It consists of 17 videos recorded by 14 different camera wearers. There are 12 categories of actions and 29 categories of objects related to actions, e.g., taking a bowl, folding bread, and placing cheese. Each ground truth action is annotated with the corresponding category as well as the starting and ending frame numbers. This annotated information is provided by the creators, together with the egocentric videos and gaze information. As in [38], we use 13 videos for training and the remaining 4 videos for testing in our experiments. The egocentric videos are mostly captured next to a table with various types of food, dishes and snacks. Thus, most of the actions are about interacting with objects. This dataset is challenging for the following reasons:

1) The length of the annotated GT actions ranges from tens to hundreds of frames, e.g., ~ 10 frames for the shortest one and ~ 340 frames for the longest one.
2) Using hands alone is not sufficient to determine whether a frame contains an action, as hands frequently appear in non-action frames, and many action frames do not involve hands. Fig. 5 shows some of these examples.
3) Two successive actions can be divided into three types: (1) nonadjacent actions: action-irrelevant frames existed between two actions; (2) adjacent actions: two actions appearing seamlessly; and (3) overlapping actions: the end of the last action overlapping with the start of the next action. The first two actions are quite common in GTEA Gaze, while the third is rare, with only one or two frames overlapping.
4) The frequencies of occurrences of different action categories are quite different, e.g., common actions such as "take" have a much higher frequency of occurrence compared with other actions. The first two rows of Table I show the statistics of GT actions in the dataset. Some categories appear quite rarely in the training videos (the first row), e.g., "fold" and "clean" only appear once. Some categories never appear in the testing videos (the second row), e.g., "fold", "press", "cut" and "clean". Moreover, the ratios of action categories are different in the training and testing videos, e.g., "take" (44.6% vs. 35.8%), "sandwich" (1.4% vs. 5.3%).

To investigate the effectiveness of our method on unseen

videos, we also evaluate TAP generation and actionness estimation on the GTEA Gaze+ dataset [38], which consists of new action categories, in Section VI-G.

B. Sample Generation for Actionness Estimation

To prepare the training and testing samples for our actionness networks, we apply the settings suggested by [14] and adopt a ratio of $1:3$ for positive (action) and negative (non-action) samples. We consider a TAP that has an IoU overlapping ratio ≥ 0.85 with any GT action as a positive sample; otherwise, it is considered as a negative sample. To ensure diversity, samples that have an IoU ≥ 0.95 with other samples are removed. For each video, we randomly generate 1,000 samples, where 250 of the samples are positive samples and 750 are negative samples. Hence, there are a total of 13,000 samples for training (13 training videos) and 4,000 samples for testing (4 testing videos). The last two rows of Table I show the statistics of the generated positive samples for the training and testing videos. Note that the numbers belonging to different categories are largely determined by the frequencies and the lengths of GT actions.

We use the same set of samples to construct the inputs to both the spatial and temporal actionness networks, as discussed in Sections IV-A and IV-B. The labels of the training samples for the three different tasks are as follows:

- **Binary classification network**: 0 for negative samples and 1 for positive samples.
- **Multi-class classification network**: 0 for non-action samples and $1 \sim 12$ for action samples (for 12 categories of actions in the GTEA Gaze dataset [38]).
- **Regression network**: The IoU overlapping ratio between the sample and the nearest GT action as the output label.

C. Training and Testing

We fine tune all the proposed spatial and temporal actionness networks on the pre-trained AlexNet [13] provided by Caffe [44]. All networks follow the architecture of AlexNet, with five convolutional layers and two fully connected layers. The settings for all layers are unchanged, except for the data layer, the first convolutional layer and the output layer. The learning rate is set to 10^{-3} (1/10 of the initial pre-training learning rate), allowing the fine tuning to adapt to the new task while not significantly affecting the initialization. In each iteration, we construct a mini-batch with 25 positive and 75 negative samples, maintaining the $1:3$ ratio as suggested by [14]. The networks converge after approximately 13,000 iterations.

During testing, actionness can be estimated based on the outputs of different tasks:

- **Binary classification network**: The output is a 2D continuous probability vector ranging between 0 and 1, and actionness is determined by the probability of the positive class.
- **Multi-class classification network**: The output is a 13D continuous probability vector ranging between 0 and 1, and actionness is determined by the summation of probabilities of all classes except the non-action class.

TABLE II
STATISTICS FOR GENERATED 10,000 SLIDING OR RANDOM WINDOWS. "CORRELATED GTS" REFER TO THE DISCRIMINATIVE GT ACTIONS DETECTED FROM THE POSITIVE SAMPLES, AS MULTIPLE POSITIVE ONES MAY CORRELATE TO THE SAME GT ACTION. "RATIO" REFERS TO THE PERCENTAGE OF CORRELATED GTS OVER ALL LABELED GTS.

Video number	2nd	3rd	5th	20th	Average
Sliding windows:					
- No. of positive samples	141	108	127	61	109.25
- No. correlated GTs	18	26	15	20	19.75
- Ratio	0.9	0.81	1.0	0.71	0.855
Random windows:					
- No. of positive samples	103	87	154	45	97.25
- No. of correlated GTs	18	22	15	16	17.75
- Ratio	0.9	0.69	1.0	0.57	0.79

- **Regression network**: The output is a single continuous value and is directly used as the actionness score to represent the similarity with any generic action.

VI. EXPERIMENTS

To evaluate this work, we apply the metric used in object proposals for TAP evaluation, i.e., a proposal is considered as correct if its IoU overlapping ratio with any GT action is higher than a certain threshold. While this threshold is typically set to 0.5 for object proposals [18], [16], we set it to a higher value of 0.85 in this work, as explained in Section IV-C. The detection rate with respect to the number of TAPs is used as the main evaluation metric for TAP detection performance. Finally, the performance of our overall framework is evaluated by the detection rate of the top ranked TAPs to study how well our TAP generation algorithm and our actionness networks can localize generic action events in the videos.

To evaluate our work, we compare the proposed method with two baseline methods, i.e., "sliding windows" and "random windows", which are widely used in object proposal detection such as in [16], [45], [46]:

- **Sliding windows**: In the GTEA Gaze dataset [38], the lengths of the GT actions range from tens to hundreds of frames. Hence, we generate 30 different lengths at each location, e.g., 10 frames, 20 frames, ..., 300 frames, with an objective of producing a total of 10,000 sliding windows for each testing video.
- **Random windows**: To ensure the diversity of randomly generated windows, we remove windows that have a high IoU overlapping ratio of 0.95 with any other generated windows. The program runs until 10,000 windows are generated for each testing video.

Table II shows the statistics of the two baseline methods on the testing videos. We can see that even with 10,000 windows generated, only 109 (for "sliding windows") and 97 (for "random windows") on average are positive samples, whose IoU overlapping ratios with any GT actions are higher than the threshold of 0.85. Note that the positive ones may still not be able to cover all the GT actions in the video despite the large number of windows, as some GT actions are extremely short, e.g., tens of frames, making it difficult to generate appropriate windows without any prior information. We can also see that the 3rd and 20th videos are more challenging than

TABLE III
COMPARISON OF THE VELOCITY-BASED AND ACCELERATION-BASED APPROACHES FOR ACTION ATOM GENERATION. THE FRACTIONS SHOW THE DETECTION RATES VS. THE NUMBER OF TAPS.

Number of TAPs	100	200	300	400	500	600	700	800	900	1,000
Velocity-based approach	0.590	0.748	0.835	0.873	0.897	0.917	0.935	0.950	0.957	0.961
Acceleration-based approach	0.519	0.664	0.748	0.788	0.817	0.843	0.867	0.888	0.902	0.913

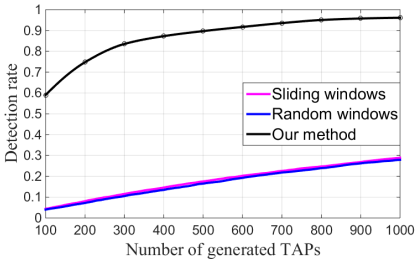


Fig. 6. Evaluation of the generated proposals. "Sliding windows" and "Random windows" are the two baseline methods. The windows generated by these methods are considered as TAPs for evaluation. "Our method" refers to the method proposed in Section III.

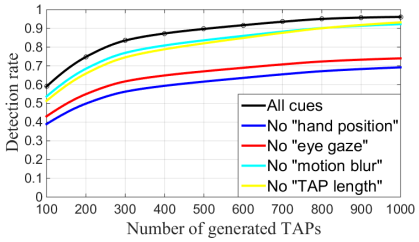
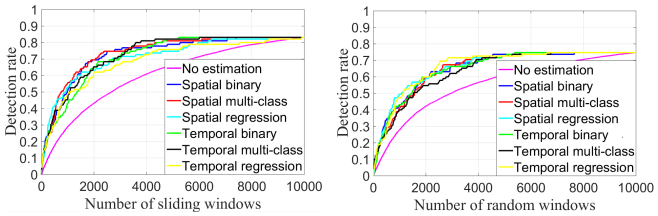


Fig. 7. Ablation study of the egocentric cues. "All cues" refers to the method proposed in Section III-B with all four egocentric cues. "No XXXX" refers to the method with one of the four cues removed.



(a) "sliding windows" (b) "random windows"

Fig. 8. Comparison of the two baseline methods with respect to the TAP detection rate of the testing videos.

the other videos. All remaining experiments on the baseline methods in this section follow the statistics shown in Table II.

For the remainder of this section, we first compare the velocity-based and acceleration-based approaches in terms of producing action atoms (Section VI-A) and our proposal generation method with the two baseline methods (Section VI-B). We then conduct an ablation study of the egocentric cues for TAP generation (Section VI-C) and evaluate the effectiveness of the spatial and temporal actionness networks (Section VI-D). We further study the overall performance and the computation time of our framework (Sections VI-E and VI-F, respectively). Finally, we apply this work to the GTEA Gaze+ dataset with new categories of actions (Section VI-G), and we discuss the feasibility of directly applying the multi-class network for action recognition (Section VI-H).

A. Action Atom Generation Comparison

In Section III-A, we have presented the velocity-based and acceleration-based approaches for segmenting a video into action atoms. In this experiment, we compare their effects on the accuracy of the resulting TAPs. Given a video, we adjust the frame-variation threshold α and the atom-variation threshold β to generate different numbers of TAPs. Table III shows the detection rates at different numbers of TAPs from averaging the results from all testing videos. Because the videos are of different lengths, we adjust α and β to generate the same number of TAPs for comparison. The results of the velocity-based approach are consistently better than those of the acceleration-based approach. This is due to the frequently irregular and non-uniform motions of the camera wearer.

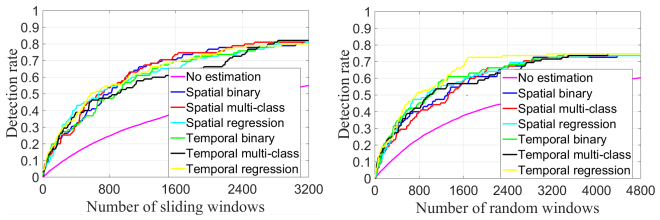
B. Proposal Evaluation

In this experiment, we study the accuracy of our generated proposals (or TAPs) without the actionness ranking step, again by adjusting the thresholds α and β to produce different numbers of TAPs. Note that the proposals generated using one set of threshold values may not have any relationship with those of another set, i.e., these proposals are not a subset of each other. Fig. 6 compares the detection rates of our method with two baseline methods, i.e., "sliding windows" and "random windows". For randomly generated windows, we perform 100 computations and report the average performance.

We can see from Fig. 6 that the proposed method achieves superior performance over the two baseline methods, while the two baseline methods achieve almost the same performance. With 100 TAPs, nearly 60% of the GTs are detected, compared with 4% for the baseline methods. With 1,000 TAPs, 96% of the GTs are detected, compared with less than 30% for the baseline methods. Such an improvement demonstrates the effectiveness of our proposal generation method based on the achieved high detection rate without actionness estimation.

C. Ablation Study

In Section III-B, we propose four egocentric cues for generating the hierarchical TAPs. To study the effectiveness of each cue, we have conducted an experiment to compare the performances of our method when all cues are used and when each of the four cues is removed. Fig. 7 shows the results of this ablation study. As expected, hand position and eye gaze have greater impacts compared with motion blur and TAP length, as the first two are more egocentric related and thus depict the intrinsic properties of an action.



(a) "sliding windows" (b) "random windows"

Fig. 9. Comparison of the two baseline methods on the TAP detection rate of the testing videos after non-maximum suppression (NMS).

D. Actionness Network Evaluation

In this experiment, we study the performances of the proposed actionness networks individually without considering TAP generation. For each video, we apply "sliding windows" and "random windows" to each of the 10,000 generated proposals as the inputs to the networks. These proposals are then ranked according to the estimated actionness values. Fig. 8 shows the detection rates with respect to different numbers of top ranked proposals. All the proposed networks achieve superior performances, compared with the two baseline methods (labeled as "no estimation"). Similar to object proposal methods, e.g., BING[16], we randomly order the proposals from the baseline methods to measure the improvement achieved by the actionness estimation. In addition, we note that the spatial networks perform slightly better than the temporal networks on the proposals using "sliding windows", while the two types of networks perform similarly to the proposals using "random windows". This is due to the distribution difference between the two types of proposals, i.e., "sliding windows" generates relatively compact proposals compared to "random windows". Because the temporal networks utilize continuous optical flow information, they are more heavily affected by the highly overlapped proposals than the spatial networks. For example, the temporal networks keep ranking similar proposals at the top, as they share similar actionness values.

To further investigate the above observation, we adopt the widely used idea from object proposals to remove redundant proposals using non-maximum suppression (NMS). For the proposals having IoU > 0.7, only the proposal with the highest actionness value is retained. In this way, many proposals are removed. Fig. 9 shows the results after NMS. We draw the following conclusions as we compare Fig. 9 with Fig. 8:

1) Different numbers of proposals are retained after NMS, i.e., 3,200 proposals for "sliding windows" and 4,800 proposals for "random windows", indicating that "sliding windows" generates highly overlapped proposals. A comparable performance is achieved compared with the original 10,000 proposals ($1\% \sim 2\%$ difference).
2) For "sliding windows", the detection rates are improved when the same numbers of top ranked proposals are used (temporal regression): 500 (32% vs. 43%), 1,000 (49% vs. 58%), and 2,000 (61% vs. 73%).
3) For "random windows", the detection rates are improved when the same numbers of top ranked proposals are used (temporal regression): 500 (30% vs. 39%), 1,000 (48% vs. 56%), and 2,000 (60% vs. 72%).
4) The temporal networks perform similarly to the spatial networks with "sliding windows" but better with "random windows".

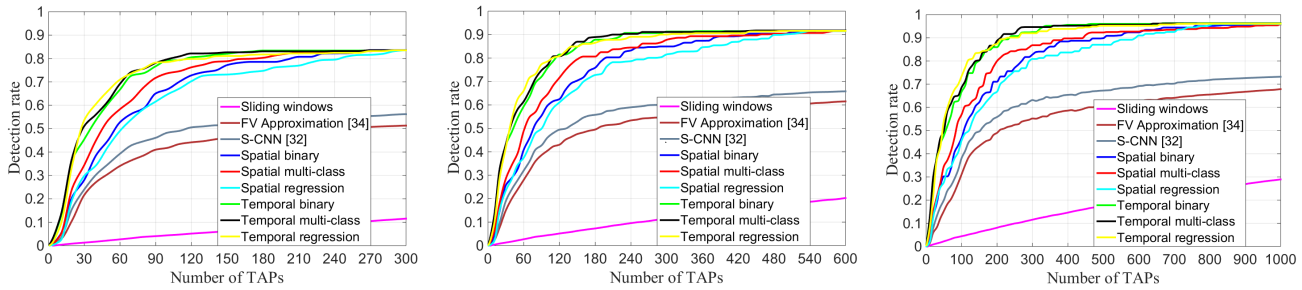
We further study these conclusions in the next experiment.

E. Overall Performance Comparison

In this experiment, we evaluate the overall performance of the proposed framework by combining TAP generation with the actionness networks. "Sliding windows" is used as the baseline method because "random windows" has a similar performance, as shown in the last experiment. Two popular works on temporal action localization, S-CNN [32] and FV Approximation [34], originally designed for hand-held untrimmed videos, are also used as baselines to investigate the feasibility of applying them to egocentric videos. Each testing video is evaluated using the following methods: "sliding window", S-CNN, FV Approximation, the spatial and temporal actionness networks. As proposed in Section IV-C, three different architectures, binary classification, multi-class classification and regression, are applied for each actionness network producing a total of 3+2*3=9 comparisons. Motivated by [18] for object proposals, three sets containing different total numbers of TAPs are evaluated. For each set, we rank the TAPs and then select different numbers of top ranked TAPs for evaluation. To provide a fair comparison, the prediction results of the localization network with post-processing are used for S-CNN, and an approximation with branch-and-bound search is used to score the proposals for FV Approximation. Fig. 10 shows the detection rates using different numbers of proposals.

The proposed framework performs significantly better than "sliding windows" and favorably against [32], [34], as a large number of their outputted candidates only slightly overlap with the GT actions. This shows that there are significant differences between the actions appearing in hand-held videos and egocentric videos, e.g., motion patterns, frequency and duration. The proposed method can detect most of the generic action events with hundreds of TAPs. The TAP sets of different sizes can benefit applications with different requirements, similar to [18]: (1) The first set with a total of 300 TAPs achieves the best detection rate when the top 100 TAPs are used: $\sim$ **80%** (Fig. 10(a)) vs. $\sim 77\%$ (Fig. 10(b)) vs. $\sim 73\%$ (Fig. 10(c)). (2) The second set with a total of 600 TAPs achieves the best detection rate when the top 200 TAPs are used: $\sim 82\%$ (Fig. 10(a)) vs. $\sim$ **90%** (Fig. 10(b)) vs. $\sim 88\%$ (Fig. 10(c)). (3) The third set with a total of 1000 TAPs achieves the best detection rate when the top 300 TAPs are used: $\sim 83\%$ (Fig. 10(a)) vs. $\sim 91\%$ (Fig. 10(b)) vs. $\sim$ **95%** (Fig. 10(c)).

We can also see that the temporal networks outperform the spatial networks on all three sets. This shows that temporal information is more important than spatial information in action recognition, and a generic action event can be recognized by a few scanlines of optical flows. This agrees with our previous observation that temporal information is more useful for proposals without significant overlapping. We further note that the multi-class classification performs slightly better among



(a) Detection rate (TAPs from a total of 300) (b) Detection rate (TAPs from a total of 600) (c) Detection rate (TAPs from a total of 1,000)

Fig. 10. Overall performance comparison. Curves of the six proposed networks are computed using the identical TAPs as inputs.

TABLE IV

EVALUATION USING MULTI-CLASS CLASSIFICATION NETWORKS DIRECTLY FOR ACTION RECOGNITION. THE SCORES SHOWN ARE THE AVERAGE CONFIDENCE VALUES OF THE CORRESPONDING CATEGORY. NOTE THAT THE LAST FOUR CATEGORIES ARE MISSING IN THE TESTING SET.

Category	Non-action	Take	Open	Scoop	Spread	Close	Put	Sandwich	Pour	Fold	Press	Cut	Clean
Spatial:													
- Training set	0.976	0.476	0.364	0.425	0.261	0.326	0.227	0.245	0.598	0.013	0.098	0.010	0.088
- Testing set	0.948	0.222	0.090	0.037	0.023	0.006	0.009	0.001	0.334	NaN	NaN	NaN	NaN
Temporal:													
- Training set	0.986	0.754	0.641	0.541	0.631	0.638	0.652	0.666	0.610	0.626	0.660	0.564	0.819
- Testing set	0.971	0.294	0.050	0.067	0.064	0.009	0.075	0.015	0.240	NaN	NaN	NaN	NaN

the three temporal networks and obviously better among the three spatial networks. This is mainly because the multi-class networks learn the explicit representations of different categories of actions, which have stronger classification capacity compared with the binary classification and regression networks, as they assume each category has distinctive motion patterns.

F. Computational Time

In this experiment, we study the time efficiency of the proposed method. We conduct the tests on a notebook with an i7 2.6 GHz CPU, a GTX970M GPU, and 16 GB of RAM. We have implemented our method on Caffe [44]. Our networks take on average $\sim 150ms$ on the GPU and $\sim 1.6s$ on the CPU to estimate the actionness of a TAP. Our algorithms take on average $\sim$ 3 minutes for TAP generation and another $\sim$ 2 minutes for actionness estimation for each egocentric video in the GTEA Gaze dataset [38] to produce approximately 300 ranked TAPs as output. Hence, it only takes $\sim$ 5 minutes for our method to reduce the number of candidates from 10,000+ to 300 with a superior detection rate (see Fig. 10), $\sim$ 7.5 minutes to 600 ranked TAPs, and $\sim$ 10 minutes to 1,000 ranked TAPs. Assuming that the action classifier takes $\sim 200ms$ to recognize the category of each input candidate, the reduced 9700+ candidates save at least 32 minutes in the recognition task. For the GTEA Gaze+ dataset [38] consisting of meal preparation videos, it takes on average $\sim$ 12 minutes for TAP generation and another $\sim$ 1.7 minutes for actionness estimation to approximately output 300 ranked proposals, as the generated TAPs before NMS are comparably fewer in number for producing approximately the same number of outputs for longer videos. Given a video, the number of output proposals does not evidently affect the time cost of TAP generation, as the scores from the proposed egocentric cues can be efficiently propagated through the TAP hierarchy. The time cost of actionness estimation is only determined by the number of TAPs and is not affected by the video or TAP length. To study the reduction in computation time for action recognition, this work is combined with [11] by replacing the sliding windows, and the efficiency is significantly improved ($\sim$ 10 fps vs. $\sim$ 45 fps) on the TrecVid MED dataset. Moreover, the combination of this work and [34] is 3.2 times faster than the original exhaustive search on the Duchenne dataset.

G. Performances on GTEA Gaze+ Dataset

To investigate the effectiveness of this work on unseen videos, we further evaluate TAP generation and actionness estimation on the GTEA Gaze+ dataset [38]. The egocentric videos are captured in an instrumented house with a kitchen that contains all standard appliances and furnishings. This dataset consists of more categories of actions compared with GTEA Gaze, and some categories are new for the actionness network, e.g., compressing, drying, and reading. The videos are mostly about meal preparation activities, and each video contains approximately 100 different actions vs. approximately 30 actions in GTEA Gaze.

Similar to the settings introduced in Section VI-E, we evaluate the proposed method by generating different numbers of TAPs: 300, 600 and 1,000. The temporal binary classification network is used for actionness estimation, and the detection rate of each set is as follows: (1) $\sim 71\%$ of actions with all 300 TAPs and $\sim 60\%$ of actions with the 100 top ranked TAPs after actionness estimation; (2) $\sim 82\%$ of actions with all 600 TAPs and $\sim 76\%$ of actions with the 200 top ranked TAPs after actionness estimation; and (3) $\sim 93\%$ of actions with all 1,000 TAPs and $\sim 90\%$ of actions with the 300 top ranked TAPs after actionness estimation. These results indicate that both TAP generation and actionness estimation work well on unseen videos with unseen actions. The performance is not

as good as in GTEA Gaze, as GTEA Gaze+ contains more categories of actions.

H. Discussion

Our earlier experiments show that our multi-class classification performs well among the spatial and temporal networks. One may then ask if we can apply multi-class classification directly for action recognition, i.e., identifying the category of a TAP instead of measuring how likely it is to contain a generic action. Hence, we conduct an experiment to study this feasibility. Table IV shows the average confidence values of the 12 action categories plus the non-action category by applying the spatial/temporal multi-class classification networks on the training and testing sets. Each video contains 1,000 samples (250 positive and 750 negative samples). We can see that the temporal network significantly outperforms the spatial network on all categories of the training set, especially for the last four categories, which are rare actions according to Table I. However, the performances of both types of networks drop dramatically on all categories of the testing set except for the "non-action" set. This indicates that the number of training samples is insufficient to train such a multi-class classifier. We have found that the limited training samples have very different motion patterns to those in the testing videos. We can also see that both "take" and "pour" actions achieve much higher performances compared with other actions. This is because the "take" action has the highest number of training samples, while the "pour" action has very similar motion patterns in both the training and testing videos.

In conclusion, existing egocentric video datasets may be too small to train a multi-class classification network for action recognition. As future work, we are currently investigating possible methods of creating a larger dataset for this purpose.

VII. Conclusions

In this paper, we have presented *temporal action proposals* (TAPs) for egocentric videos. Each TAP is a video clip extracted from the input egocentric video and may contain a generic action performed by the camera wearer. The motivation for extracting TAPs from the egocentric video is to accelerate the action recognition task by replacing the commonly used sliding window strategy. We have proposed two techniques to extract TAPs: action atom generation and hierarchical clustering. We have also investigated the spatial and temporal networks for actionness estimation to rank the TAPs. Extensive evaluations show that our proposals, i.e., TAPs, and our actionness networks achieve promising performances.

References

[1] Y. Li, A. Fathi, and J. Rehg, "Learning to predict gaze in egocentric video," in *ICCV*, 2013, pp. 3216–3223.

[2] Y. Li, Z. Ye, and J. Rehg, "Delving into egocentric actions," in *CVPR*, 2015, pp. 287–295.

[3] B. Xiong and K. Grauman, "Detecting snap points in egocentric video with a web photo prior," in *ECCV*, 2014, pp. 282–298.

[4] R. Yonetani, K. Kitani, and Y. Sato, "Recognizing micro-actions and reactions from paired egocentric videos," in *CVPR*, 2016, pp. 2629–2638.

[5] Y. Hoshen and S. Peleg, "An egocentric look at video photographer identity," in *CVPR*, 2016, pp. 4284–4292.

[6] H. Park, J. Hwang, Y. Niu, and J. Shi, "Egocentric future localization," in *CVPR*, 2016, pp. 4697–4705.

[7] N. Rhinehart and K. Kitani, "Learning action maps of large environments via first-person vision," in *CVPR*, 2016, pp. 580–588.

[8] D. Jayaraman and K. Grauman, "Learning image representations tied to ego-motion," in *ICCV*, 2015, pp. 1413–1421.

[9] P. Siva and T. Xiang, "Weakly supervised action detection," in *BMVC*, vol. 2, no. 4, 2011, p. 6.

[10] T. Lan, Y. Wang, and G. Mori, "Discriminative figure-centric models for joint action localization and recognition," in *ICCV*, 2011, pp. 2003–2010.

[11] D. Oneata, J. Verbeek, and C. Schmid, "Action and event recognition with fisher vectors on a compact feature set," in *ICCV*, 2013, pp. 1817–1824.

[12] H. Wang, D. Oneata, J. Verbeek, and C. Schmid, "A robust and efficient video representation for action recognition," *IJCV*, pp. 219–238, 2016.

[13] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

[14] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014, pp. 580–587.

[15] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, "Selective search for object recognition," *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.

[16] M. Cheng, Z. Zhang, W. Lin, and P. Torr, "Bing: Binarized normed gradients for objectness estimation at 300fps," in *CVPR*, 2014, pp. 3286–3293.

[17] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *CVPR*, 2014, pp. 328–335.

[18] C. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014, pp. 391–405.

[19] S. He and R. Lau, "Oriented object proposals," in *ICCV*, 2015, pp. 280–288.

[20] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *TPAMI*, 2015.

[21] F. Perazzi, O. Wang, M. Gross, and A. Sorkine-Hornung, "Fully connected object proposals for video segmentation," in *ICCV*, 2015, pp. 3227–3234.

[22] G. Yu and J. Yuan, "Fast action proposals for human action detection and search," in *CVPR*, 2015, pp. 1302–1311.

[23] C. Li and K. Kitani, "Model recommendation with virtual probes for egocentric hand detection," in *ICCV*, 2013, pp. 2624–2631.

[24] S. Bambach, S. Lee, D. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *ICCV*, 2015, pp. 1949–1957.

[25] Y. Zhou, B. Ni, R. Hong, X. Yang, and Q. Tian, "Cascaded interactional targeting network for egocentric video analysis," in *CVPR*, 2016, pp. 1904–1913.

[26] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *CVPR*, 2016, pp. 1933–1941.

[27] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *ECCV*, 2010, pp. 282–295.

[28] M. Ma, H. Fan, and K. M. Kitani, "Going deeper into first-person activity recognition," in *CVPR*, 2016, pp. 1894–1903.

[29] S. Singh, C. Arora, and C. Jawahar, "First person action recognition using deep learned descriptors," in *CVPR*, 2016, pp. 2620–2628.

[30] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *ICCV*, 2015, pp. 3164–3172.

[31] A. Gaidon, Z. Harchaoui, and C. Schmid, "Temporal localization of actions with actoms," *TPAMI*, vol. 35, no. 11, pp. 2782–2795, 2013.

[32] Z. Shou, D. Wang, and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage CNNs," in *CVPR*, 2016, pp. 1049–1058.

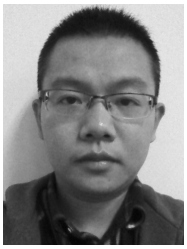
[33] J. Yuan, B. Ni, X. Yang, and A. A. Kassim, "Temporal action localization with pyramid of score distribution features," in *CVPR*, 2016, pp. 3093–3102.

[34] D. Oneata, J. Verbeek, and C. Schmid, "Efficient action localization with approximately normalized fisher vectors," in *CVPR*, 2014, pp. 2545–2552.

[35] T. Brox and J. Malik, "Large displacement optical flow: descriptor matching in variational motion estimation," *TPAMI*, vol. 33, no. 3, pp. 500–513, 2011.

[36] A. Fathi, X. Ren, and J. Rehg, "Learning to recognize objects in egocentric activities," in *CVPR*, 2011, pp. 3281–3288.

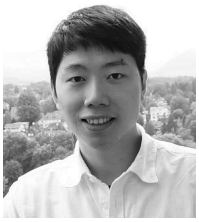
[37] C. Li and K. Kitani, "Pixel-level hand detection in ego-centric videos," in *CVPR*, 2013, pp. 3570–3577.
[38] A. Fathi, Y. Li, and J. Rehg, "Learning to recognize daily actions using gaze," in *ECCV*, 2012, pp. 314–327.
[39] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.
[40] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," in *ICCV*, 2011, pp. 2564–2571.
[41] F. Crete, T. Dolmiere, P. Ladret, and M. Nicolas, "The blur effect: perception and estimation with a new no-reference perceptual blur metric," in *Electronic Imaging*, 2007.
[42] W. Chen, C. Xiong, R. Xu, and J. Corso, "Actionness ranking with lattice conditional ordinal random fields," in *CVPR*, 2014, pp. 748–755.
[43] Y. Luo, L.-F. Cheong, and A. Tran, "Actionness-assisted recognition of actions," in *ICCV*, 2015, pp. 3244–3252.
[44] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM Multimedia*, 2014, pp. 675–678.
[45] B. Alexe, T. Deselaers, and V. Ferrari, "What is an object?" in *CVPR*, 2010, pp. 73–80.
[46] Y. Lee and K. Grauman, "Learning the easy things first: Self-paced visual category discovery," in *CVPR*, 2011, pp. 1721–1728.



Shao Huang is a joint Ph.D. student of University of Chinese Academy of Sciences and City University of Hong Kong. He obtained his B.E. degree from Nankai University in 2011. His research interests include multimedia technology, pattern recognition, image processing and computer vision.

Weiqiang Wang received the B.E. and M.E. degrees in computer science from Harbin Engineering University, Harbin, China, in 1995 and 1998, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, in 2001.

He is currently a Professor with the Graduate School, CAS, and a Guest Researcher in ICT, CAS. His research interests include multimedia content analysis and computer vision.



Shengfeng He is an Associate Professor in the School of Computer Science and Engineering at South China University of Technology. He obtained his B.Sc. degree and M.Sc. degree from Macau University of Science and Technology in 2009 and 2011 respectively, and the Ph.D degree from City University of Hong Kong in 2015. His research interests include computer vision, image processing, computer graphics, and deep learning.

Rynson W.H. Lau received his Ph.D. degree from University of Cambridge. He was on the faculty of Durham University and is now with City University of Hong Kong. Rynson serves on the Editorial Boards of Computer Graphics Forum, and Computer Animation and Virtual Worlds. He has served as a Guest Editor of a number of journal special issues, including ACM Trans. on Internet Technology, IEEE Trans. on Multimedia, IEEE Trans. on Visualization and Computer Graphics, and IEEE Computer Graphics & Applications. He has also served in the committee of a number of conferences, including Program Co-chair of ACM VRST 2004, ACM MTDL 2009, IEEE U-Media 2010, and Conference Co-chair of CASA 2005, ACM VRST 2005, ACM MDI 2009, ACM VRST 2014. Rynson's research interests include computer graphics and computer vision.