

Region-sequence based six-stream CNN features for general and fine-grained human action recognition in videos

Ma, Miao; Marturi, Naresh; Li, Yibin; Leonardis, Ales; Stolkin, Rustom

DOI:

[10.1016/j.patcog.2017.11.026](https://doi.org/10.1016/j.patcog.2017.11.026)

License:

Creative Commons: Attribution-NonCommercial-NoDerivs (CC BY-NC-ND)

Document Version

Peer reviewed version

Citation for published version (Harvard):

Ma, M, Marturi, N, Li, Y, Leonardis, A & Stolkin, R 2018, 'Region-sequence based six-stream CNN features for general and fine-grained human action recognition in videos', *Pattern Recognition*, vol. 76, pp. 506-521. <https://doi.org/10.1016/j.patcog.2017.11.026>

[Link to publication on Research at Birmingham portal](#)

Publisher Rights Statement:

Checked for eligibility: 29/11/2017

General rights

Unless a licence is specified above, all rights (including copyright and moral rights) in this document are retained by the authors and/or the copyright holders. The express permission of the copyright holder must be obtained for any use of this material other than for purposes permitted by law.

- Users may freely distribute the URL that is used to identify this publication.
- Users may download and/or print one copy of the publication from the University of Birmingham research portal for the purpose of private study or non-commercial research.
- User may use extracts from the document in line with the concept of 'fair dealing' under the Copyright, Designs and Patents Act 1988 (?)
- Users may not further distribute the material nor use it for the purposes of commercial gain.

Where a licence is displayed above, please note the terms and conditions of the licence govern your use of this document.

When citing, please reference the published version.

Take down policy

While the University of Birmingham exercises care and attention in making items available there are rare occasions when an item has been uploaded in error or has been deemed to be commercially or otherwise sensitive.

If you believe that this is the case for this document, please contact UBIRA@lists.bham.ac.uk providing details and we will remove access to the work immediately and investigate.

Accepted Manuscript

Region-sequence based six-stream CNN features for general and fine-grained human action recognition in videos

Miao Ma, Naresh Marturi, Yibin Li, Ales Leonardis, Rustam Stolkin

PII: S0031-3203(17)30478-8
DOI: [10.1016/j.patcog.2017.11.026](https://doi.org/10.1016/j.patcog.2017.11.026)
Reference: PR 6379



To appear in: *Pattern Recognition*

Received date: 4 May 2017
Revised date: 31 October 2017
Accepted date: 19 November 2017

Please cite this article as: Miao Ma, Naresh Marturi, Yibin Li, Ales Leonardis, Rustam Stolkin, Region-sequence based six-stream CNN features for general and fine-grained human action recognition in videos, *Pattern Recognition* (2017), doi: [10.1016/j.patcog.2017.11.026](https://doi.org/10.1016/j.patcog.2017.11.026)

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

Region-sequence based six-stream CNN features for general and fine-grained human action recognition in videos

Miao Ma^{1,2,3,†}, Naresh Marturi^{2,4}, Yibin Li³, Ales Leonardis², and Rustam Stolkin²

¹Qingdao University, Qingdao, Shandong, 266071, PR China;

²University of Birmingham, Edgbaston, Birmingham, B15 2TT, UK;

³Shandong University, Jinan, Shandong, 250061, PR China;

⁴KUKA robotics UK Ltd., Wednesbury Great Western Street, WS10 7LL, UK;

[†]Corresponding author: mxm443@cs.bham.ac.uk

Abstract

This paper addresses the problems of both general and also fine-grained human action recognition in video sequences. Compared with general human actions, fine-grained action information is more difficult to detect and occupies relatively small-scale image regions. Our work seeks to improve fine-grained action discrimination, while also retaining the ability to perform general action recognition. Our method first estimates human pose and human parts positions in video sequences by extending our recent work on human pose tracking, and crops different scaled patches to obtain richer action information in a variety of different scales of appearance and motion cues. We then utilize a Convolutional Neural Network (CNN) to process each such image patch. Instead of using the output one dimension feature from the full-connection layer, we utilize the outputs of the pooling layer of CNN structure, which contains more spatial information. Then the high dimension of the pooling features is reduced by encoding, to generate the final human action descriptors for classification. Our method reduces feature dimension while also effectively combining appearance and motion information in a unified framework. We have carried out empirical experiments using two publicly available human action datasets, comparing the human action recognition result of our algorithm against six recent state-of-the-art methods from the literature. The results suggest comparatively strong performance of our method.

Keywords: human pose, action recognition, video understanding

1. Introduction

Video sequences provide much richer information about actions, compared to an individual still image. Consider a single still image showing a man holding a knife in a kitchen scene. We cannot tell what he is doing with the knife. Does he want to cut something? Or is he cleaning the knife? In contrast, such action understanding is often much more easily obtainable from a video sequence. Still images provide spatial cues [1] and provide information to answer “what is that?”; while video provides both spatial and temporal cues [2], and can answer “what is going on?”.

8 Recognition of human-induced actions in videos has gained significant amount of interest in the fields of
9 computer vision and pattern recognition. This is due to its increasingly large number of applications in the
10 areas of human-machine interaction [3], intelligent space[4], virtual reality, elderly care [5], robotics [6] etc.
11 It also plays a vital role in many computer vision tasks such as video annotation, video retrieval etc. Human
12 action recognition, which is a task of assigning videos to a set of action classes, is a challenging problem due
13 to: large variety of activities [7], complex human actions and background movements [8], various observation
14 views [2] and limited observation capacities [9], as well as ambiguous movements of different actions [10]. It
15 has been intensively studied in the literature for more than two decades, where the corresponding methods
16 varies from template matching [11], hand-crafted points of interest features [8], to deep learning methods [12].

17 While a variety of methods have been proposed in the recent years to recognize actions, most of them
18 focus on videos with coarse actions [13, 14], such as lifting: where upper body moves upward; diving: where
19 entire body drops; kicking: where one leg moves while the other remains static; etc. However, these are not
20 typical in general scenes of life. In many applications, fine-grained actions [7] need to be recognized, e.g.
21 washing hands versus falling water from tap. In this paper, we primarily focus on this problem of recognising
22 fine-grained actions in the videos. It is highlighted that, for fine-grained action recognition, spatial regions
23 that contain contextual cues have most distinguished information and should receive greater attention. Some
24 related works in the literature [15, 16] use thousands of region proposals to extract action information, and
25 then choose the most distinguished region features for action recognition. However, these methods can suffer
26 in cluttered scenes, e.g. an image of a person squeezing an orange in the kitchen, with a stirrer nearby, might
27 be erroneously recognised as stirring rather than squeezing. We estimate coarse pose along videos, and then
28 extract the human body region and operation region in each frame, in order to enhance the effective pixels
29 for human action. Then the enhanced patches are processed with CNNs.

30 CNN structure contains convolutional layers, pooling layers, and fully connected layers. In convolutional
31 and corresponding pooling layers, the kernel traverse all over the image with specified size and stride, which
32 makes the output data maintain some spatial information. However, fully connected layer transforms previ-
33 ous multi-dimension data into one dimension vector, which would change and loose the spatial information
34 mentioned in previous layers. That is to say, the last pooling layer data of CNN structure contains more
35 spatial information than generally used full connection layer data. In our work, we propose an effective
36 encoding method for the pooling layer data, which is able to make better use of spatial information and help
37 to obtain more distinguishable descriptors for general and fine-grained human actions.

38 *1.1. Overview of our method*

39 In this section we provide an overview of the proposed method, highlighting important steps. Given a
40 video, the following steps are performed to recognise fine-grained actions: firstly, we introduce a pose esti-
41 mation method, which evaluates pose candidates based on appearance information and motion information,

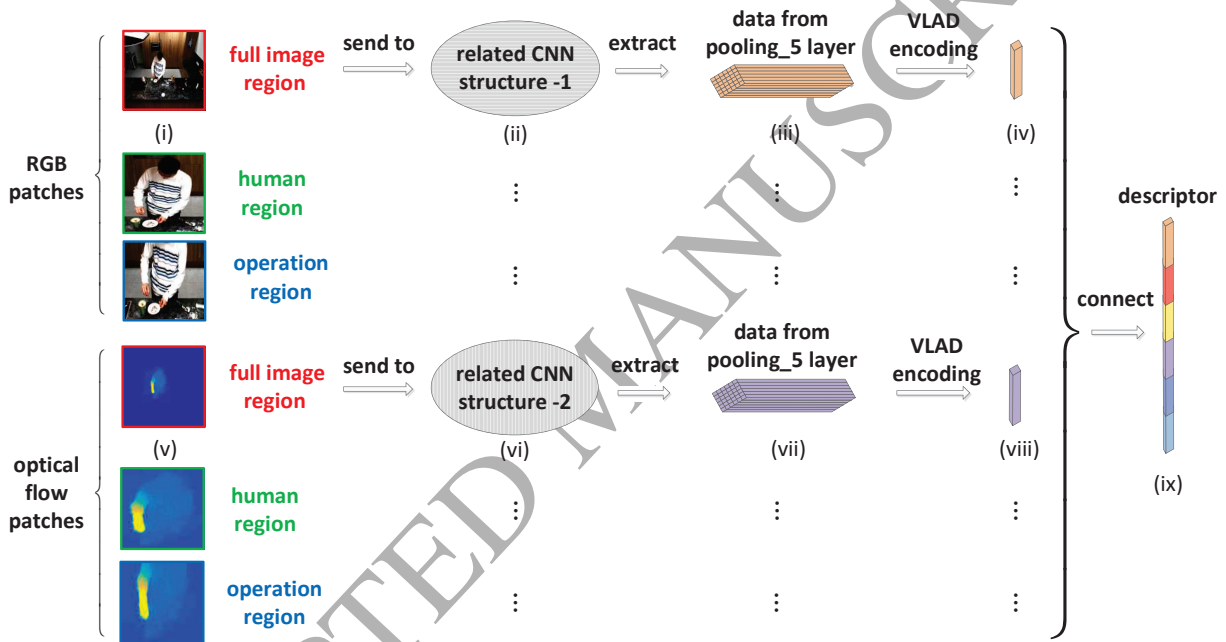
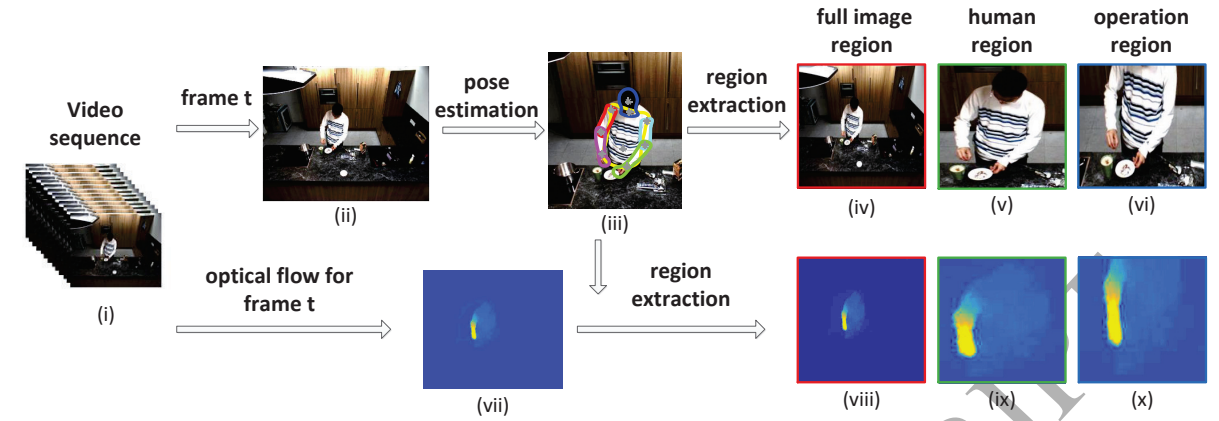


Figure 1: Illustration of the proposed method pipeline. (a) Raw video sequence as system input, and then human pose detector is used for each frame to help segment all image region, human region and operation region patches for both color images and optical flow images. (b) Each image patch is processed by related CNN structure, and data from *pooling₅* layer are encoded to obtain final fine-grained action descriptor.

42 and then we use the human pose estimation method to obtain the human body positions and regions in each
 43 video frame. Secondly, six image patch sequences are obtained based on the human body positions, and
 44 these patches are later fed to CNN structures as inputs. Finally, we construct human action discriminative
 45 features for videos by encoding pooling layer outputs of the CNN structures.

46 A detailed pipeline of our method is shown in Fig. 1. The main contributions of this work are explained
 47 as follows:

- 48 (i) Firstly, we propose a coarse human pose tracking and estimation method by extending our recent work
 49 on human pose tracking [17], see steps (i), (ii) and (iii) of Fig. 1(a). Unlike in our previous work, for
 50 recognizing human actions, we are not interested in the very exact locations of human keypoints; but
 51 aim at obtaining human body foreground regions and the corresponding appearance and motion cues
 52 to distinguish different human actions. In this case, our proposed human coarse pose tracker focuses
 53 on achieving the continuity and consistency of human foreground region in the video.
- 54 (ii) Secondly, we propose a method to use six image patch sequences to enhance and extract different scales
 55 and types of information for both appearance and motion cues. RGB and optical flow image sequences
 56 are used for obtaining appearance and motion information separately. The tracked human poses (from
 57 step (iii) of Fig. 1(a)) are used to crop regions from both RGB and optical flow images (see steps (ii)
 58 and (vii) of Fig. 1(a)). For each type of image, we get patches that contain human foreground regions
 59 (see steps (v) and (ix) of Fig. 1(a)), and regions around human arms (see steps (vi) and (x) of Fig. 1(a)).
 60 As a result, six image patch sequences are obtained for each video sequence. Each image patch has
 61 been resized to be of the same size *i.e.*, 224×224 pixels. The obtained patches enhance and insert
 62 effective pixels for recognizing fine-grained human actions.
- 63 (iii) Thirdly, instead of using the fully connected layer outputs of CNN structures, we propose a feature
 64 constructing method by encoding the outputs of last pooling layers (see steps (iii) and (vii) of Fig. 1(b))
 65 using the vector of locally aggregated descriptors (VLAD) encoding method (see steps (iv) and (viii) of
 66 Fig. 1(b)). The last pooling layer data contain more spatial information than that of the fully connected
 67 layer, and some of the spatial information would be lost in the following full connection operation.
 68 Consequently, in our proposed method, we utilize data and information from the last pooling layer for
 69 patches of different types and scales. The proposed encoding and assembling method makes better use
 70 of multiple types of action information.

71 We test our method on two publicly available datasets: sub-JHMDB dataset and MPII Cooking dataset,
 72 and the results are compared against six other state-of-the-art algorithms.

73 The remainder of the paper is organized as follows. The methods that are closely related to our work are
 74 presented in Section 2. The proposed method for obtaining contextual cue regions is presented in Section 3.
 75 Experiments performed to validate the proposed method are discussed in Section 4. Section 5 provides
 76 concluding remarks and suggestions for future work.

77 2. Related Work

78 Human action recognition is a key research area in the field of computer vision, and has been previously
 79 surveyed by many researchers [18, 19]. In this section, we discuss some of the most relevant related work.

80 In the past decade, local features such as SIFT [20], HOG [21], HOF [22] etc. have been widely used
81 for accomplishing visual recognition tasks [23, 24]. These methods often firstly extract spatial and temporal
82 local features, and then use encoding methods, for example, bag of features (BoF) [25] to encode local
83 features into vectored collections, and finally a classifier such as SVM is used for classification. However,
84 each kind of feature is only able to describe a single property, such as color, contour or salient points. As a
85 result, researchers rely more and more on combining several features together [26, 27] to represent complex
86 properties. For instance, from their survey on multi-view learning, Xu *et al.* [28] identified that multi-
87 view learning is rendered more effectively by exploring the consistency and complementary properties of
88 different views. Later in [29], they handled the incomplete-view problem for image restoration by exploiting
89 the connections between multiple views. Similarly, Li *et al.* [30] solved the image re-ranking problem by
90 exploiting the complementarity between the deep features and shallow representations, and by integrating
91 these two heterogeneous features into a multi-view feature learning model. Furthermore, Bregonzio *et al.* [31]
92 fuse local space and time individual descriptors with global spatio-temporal distribution information to solve
93 action recognition problem. It is clear that these combined features improve the visual recognition accuracy
94 compared with single feature methods.

95 In other cases, hierarchical structures [32, 33, 34] have been studied for creating more sophisticated hand-
96 crafted features. For instance, Ma *et al.* [33] proposed a hierarchical structure for action recognition, where
97 video segmentation trees are computed in the first layer using Ultrametric Contour Map (UCM) [35]. These
98 trees are then pruned in the second layer using shape, color, motion, and other information. Later in the
99 third layer, the method tracks the remaining segment trees both forwards and backwards in time, and used
100 the bag-of-words representation for recognition. Liu *et al.* [36] also proposed a hierarchical structure for
101 action recognition. In the first layer of their method, they combine optical flow with a biologically inspired
102 feature to create a distinguishable feature, which is denoted as Pyramidal Motion Feature (PMF); then in
103 the second layer, the PMFs are combined with spatial information to obtain final action descriptor.

104 Convolutional Neural Networks (CNNs) firstly emerged with the proposition of visual nervous system
105 by Hubel and Wiesel [37], and were later implemented by Fukushima [38]. Since then, many researchers
106 made efforts to improve the performance of CNN structures for various tasks. Wu *et al.* [39] proposed a
107 kind of quantized CNN (Q-CNN) framework, which simultaneously speeds up the computation and reduces
108 the storage and memory overhead of CNN models. It uses an effective training scheme to suppress the
109 accumulative error while quantizing the whole convolutional network. Ijjina *et al.* [40] proposed a genetic
110 algorithm based CNN (GA-CNN) structure, which uses solutions generated by genetic algorithms to initialize
111 the weights of CNN classifier. The proposed GA-CNN structure is able to combine the global optimization
112 capabilities of genetic algorithms with the local optimization ability of gradient descent algorithm, and as a
113 result, it minimizes the classification error and improve the performance. Wang *et al.* [41] proposed a CNN

114 packing framework (CNNpack), which handles convolutional filters in the frequency domain using discrete
115 cosine transform to compress neural networks. The CNNpack has high compression ratio and speed-up ratio
116 proofed by experiments, which creates a bridge to link traditional signal and image compression with CNN
117 compression theory.

118 Recently, CNNs have been used in a wide range of applications in different fields including computer
119 vision, e.g. object detection [15, 42] and image classification [43]. Girshick *et al.* [15] constructed region-
120 based CNN (R-CNN) features for detecting objects in still images, and the regions are pre-detected multiple
121 parallel regions which are possible to contain distinguishable object information. Later, Gkioxar *et al.* [16]
122 extended multiple parallel regions in [15] to become one primary region which is detected by a human
123 detector and a set of secondary regions for human action recognition problem in static images. Lin *et al.* [44]
124 proposed a bilinear CNN model for fine-grained object recognition in still images. In their work, they have
125 used two similar CNN architectures for one RGB frame and calculated the outer product of the two CNN
126 output matrices to obtain a vector descriptor for object recognition in static images. Yang *et al.* [3] made
127 use of a two-stream CNN structure for recognizing human grasp actions in videos. In that work, one CNN
128 stream was used for classifying the hand grasp type and the other for object recognition. Karpathy *et al.* [12]
129 used a two stream CNN structure for video classification: one stream for low-resolution images and another
130 for high-resolution center regions. Simonyan and Zisserman [2] extended the two stream CNN structure for
131 action recognition in videos, but used one stream for color images and the other for optical flow images. This
132 structure improved the action recognition accuracy by making use of both spatial and temporal information.
133 Cheron *et al.* [10] extracted image patches around estimated human pose joints for CNN processing, showing
134 that knowledge of human pose contains useful information about human actions. They also revealed that the
135 accuracy of detected human poses plays an important role. However, human pose estimation and tracking
136 is another challenging problem in its own right [17].

137 Inspired by region-based methods for still images [15, 16], we propose a region-based action recognition
138 method for videos. We propose a coarse human pose detector, which estimates human poses in each frame
139 of the video to identify human foreground regions. Additionally, we define an operation region for humans
140 who are performing fine-grained actions. Heretofore, we achieve six-stream patch sequences that are then
141 processed by the CNN structures. We use two types of CNN structures: one for the three RGB streams,
142 and the other for the three optical flow streams. The three RGB streams represent the pyramid appearance
143 cues whereas the three optical flow streams represent the pyramid motion cues. The last CNN pooling layer
144 outputs are then processed by encoding to generate final video descriptors, which are able to recognize a
145 variety of fine-grained human actions.

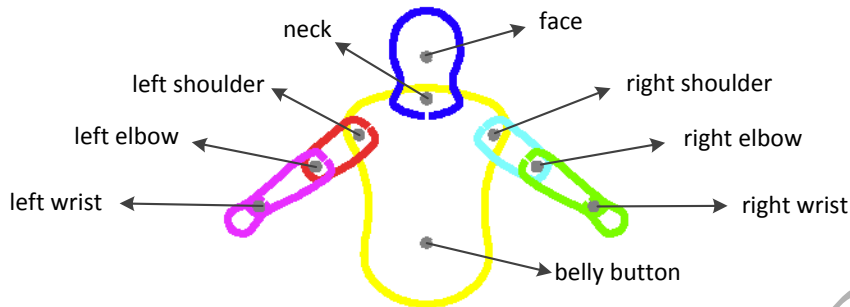


Figure 2: A sample DS puppet model for the human upper body. Colored wireframes represent different body parts and grey points represent the corresponding keypoints.

146 3. Method

147 The procedure and pipeline of our proposed method is illustrated in Fig. 1. In this section we introduce
148 them in detail.

149 3.1. Human pose estimation in video sequence

150 We build on our recently published state-of-the-art human pose tracker [17] to track human poses,
151 yielding information-rich features which can then be fed into the CNN architecture. The utilized DS puppet
152 model [45] (see Fig. 2) is a part-based probabilistic model, which represents human body parts as their
153 natural shapes, connected in a kinematic chain. This model is learned from training contours derived from
154 SCAPE [46], which is a 3D model for articulated human shape. The variability of the model parts is gained
155 through Principle Component Analysis (PCA). The DS puppet model and the corresponding keypoints are
156 illustrated in Fig. 2.

157 Human body pose candidates are initialized by utilizing the method of flowing DS puppets [47], which
158 proposes DS pose candidates through the entire video. With image evidences from adjacent frames propa-
159 gated forwards and backwards over time, each frame obtains much richer cues for consistent pose estimation.

160 In this case, for each frame t in a video sequence $\{t|t \in [1, 2, \dots, T]\}$, we have body poses generated for
161 the frame t as well as body poses propagated from frame $t - 1$ and $t + 1$. The body pose is evaluated by
162 a contour-based term $p_{ct}(I_t|\mathbf{x}_t)$, a color based term $p_{cl}(I_t|\mathbf{x}_t)$, and a hand likelihood term $p_h(I_t|\mathbf{x}_t)$ [17], as
163 shown in Eq. (1):

$$p(I_t|\mathbf{x}_t) = \lambda_{ct}p_{ct}(I_t|\mathbf{x}_t) + \lambda_{cl}p_{cl}(I_t|\mathbf{x}_t) + \lambda_h p_h(I_t|\mathbf{x}_t). \quad (1)$$

164 where, I_t means the image of frame t in the video sequence, and $\mathbf{x}_t = [\mathbf{k}_t, s_t]$ represents the vector of DS
165 model variables. \mathbf{k}_t represents body pose keypoints location vector, which contains 2D location information

166 of nine elements, *i.e.* belly button, face, neck, left/right shoulders, left/right elbows as well as left/right
 167 wrists. s_t means the scale which used to fit human pose models into certain images of various size. λ_{ct} , λ_{cl}
 168 and λ_h are fixed coefficients whose values and the selection criteria are shown in Table 1.

169 Using the cost function given by Eq. (1), we are able to select one pose candidate with the highest score
 170 for each frame, but the consistency throughout the entire video sequence cannot be ensured. Besides, in
 171 our case *i.e.* for the problem of human action recognition in videos, we only need to use the human pose
 172 estimation outcome to extract various types of image patches which contain multiple action information.
 173 That is to say, in our specific work, coarse but consistent human pose results are enough, and there are no
 174 needs to seek the exact positions of each key point of the human body.

175 In order to assure the consistency of human pose estimation, we add an additional term called *consistency*
 176 *penalty term* to the cost function Eq. 1. The penalty term, denoted by $p_{cs}(I_{t-1}, I_t, I_{t+1} | \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1})$, helps
 177 filtering inconsistent pose candidates in the video sequence.

178 In frame t ($1 < t < T$), the estimated pose candidate variable \mathbf{x}_t is then propagated to frame $t + 1$
 179 using the optical flow affine matrix to get a pose candidate variable $\hat{\mathbf{x}}_{t+1} = [\hat{\mathbf{k}}_{t+1}, \hat{s}_{t+1}]$. Similarly, \mathbf{x}_t is also
 180 propagated to frame $t - 1$ to get $\hat{\mathbf{x}}_{t-1} = [\hat{\mathbf{k}}_{t-1}, \hat{s}_{t-1}]$. As a result, we are able to calculate two distance terms:

$$d_{t,t-1} = \sum_{i=1}^N |\hat{\mathbf{k}}_{t-1}(i) - \mathbf{k}_{t-1}(i)|; \quad d_{t,t+1} = \sum_{i=1}^N |\hat{\mathbf{k}}_{t+1}(i) - \mathbf{k}_{t+1}(i)|. \quad (2)$$

181 At an initial attempt, we chose the coarse consistency penalty term to be the average of two distance
 182 terms. However, we found that this would lead to over-penalty when the optical flow matrix was not reliable.
 183 To resolve this problem, we compute a summarized distance d_c between neck and face, neck and left shoulder,
 184 and neck and right shoulder. By selecting $d_c/2$ as a threshold, we define the consistency penalty term to be:

$$p_{cs}(I_{t-1}, I_t, I_{t+1} | \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}) = \begin{cases} w \times (d_{t,t-1} + d_{t,t+1})/2, & \text{if } < d_c/2 \\ d_c/2 & , \text{ otherwise} \end{cases} \quad (3)$$

185 where, w is a weight coefficient. In our implementation, we set this value to be $w = 2$ found by trial and
 186 error.

187 For the frame $t = 1$, the consistency penalty term is defined as:

$$p_{cs}(I_{t-1}, I_t, I_{t+1} | \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}) = p_{cs}(I_t, I_{t+1} | \mathbf{x}_t, \mathbf{x}_{t+1}) = \begin{cases} w \times d_{t,t+1}, & \text{if } < d_c/2 \\ d_c/2 & , \text{ otherwise} \end{cases} \quad (4)$$

188 while for the frame $t = T$, the consistency term is defined as:

$$p_{cs}(I_{t-1}, I_t, I_{t+1} | \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}) = p_{cs}(I_{t-1}, I_t | \mathbf{x}_{t-1}, \mathbf{x}_t) = \begin{cases} w \times d_{t,t-1}, & \text{if } < d_c/2 \\ d_c/2, & \text{otherwise} \end{cases} \quad (5)$$

189 Then the cost function for evaluating consistent human poses in video sequences for each frame is defined
190 as:

$$s = \lambda_p p(I_t | \mathbf{x}_t) + \lambda_{cs} p_{cs}(I_{t-1}, I_t, I_{t+1} | \mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_{t+1}) \quad (6)$$

191 where, $p(I_t | \mathbf{x}_t)$ is the cost function for human part pose candidates in each image frame as illustrated in
192 Eq. (1). λ_p and λ_{cs} are fixed coefficients, which are described in Table 1.

193 The parameter values used to test the method and their corresponding selection criteria are summarized
194 in Table 1 based on their applications and magnitudes. The values of the parameters reported in Table 1
195 are fixed for all our experiments *i.e.* for all video sequences of various datasets.

Table 1: List of the parameters used in the experiments and corresponding selection criteria.

Equation	Coefficients	Selection
Cost function for each image Eq. (1)	$\lambda_{ct} = 4, \lambda_{cl} = 1, \lambda_h = 1$	$0 < \lambda_h \leq \lambda_{cl} < \lambda_{ct}$
Cost function for video sequence Eq. (6)	$\lambda_p = 1, \lambda_{cs} = -3$	$\lambda_{cs} < 0 < \lambda_p$

196 3.2. Obtaining Contextual Cues Regions

197 Information presented by a video sequence is often divided into spatial and temporal information. The
198 spatial information exists in each individual frame, *i.e.* scenes and objects; while the temporal information
199 lies within the motion between adjacent frames, *i.e.* the movement of observers or cameras and the motion
200 of objects in the scene. In order to obtain rich temporal information, in addition to the original RGB video
201 frame images, we also calculate the optical flow images for each pair of adjacent frames.

202 3.2.1. Appearance cues

203 For any given video frame, we first extract the pose of the human actor using our method described
204 in Section 3.1, for example, as shown in Fig. 3(a). Next, for the given frame, we use the obtained pose
205 information to extract three different area patches: full image area (red), human area (green) and operation
206 area (blue), as shown in Fig. 3(b). Full image patches are obtained by resizing the original video frames to
207 be 224×224 .

208 Using the human pose estimation method illustrated in Section 3.1, we are able to obtain points of entire
209 human pose contours. The top left and bottom right points of the contours are calculated and are represented

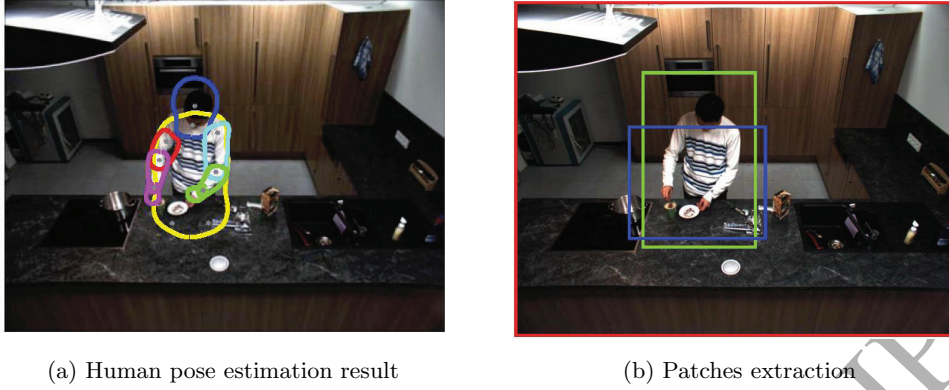


Figure 3: Crop the raw video frame image utilizing the pose estimation result to get different types of patches. (a) the human estimation result for each video frame. (b) different color boxes represents the crop edges: red box is for the full image patch, green box represents the body patch, while blue box means the operation patch.

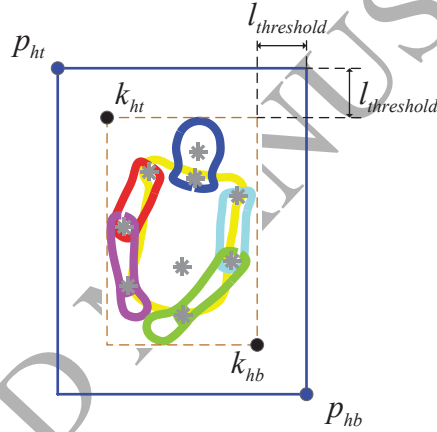


Figure 4: Calculation and selection of the body area.

210 as k_{ht} and k_{hb} respectively, as illustrated in Fig. 4. By assuming the size of the original video image frame
 211 t as $[a \times b]$, we define and set a patch margin threshold $l_{threshold}$ as shown in Eq. (7).

$$l_{threshold} = \frac{\min(a, b)}{10} \times s_t \quad (7)$$

212 where, s_t represents the scale used to fit human pose models into certain images of various size (see expla-
 213 nation below Eq. (1)).

214 Using k_{ht} , k_{hb} , and $l_{threshold}$, we are able to calculate the top left point p_{ht} and the bottom right point
 215 p_{hb} of the human body area, as defined in Eq. (8) and shown in Fig. 4. The corresponding human body

216 patch is obtained by resizing the region among p_{ht} and p_{hb} to be 224×224 , as shown in Fig. 6(b).

$$\begin{cases} p_{ht}(x) = k_{ht}(x) - l_{threshold}, & \begin{cases} p_{hb}(x) = k_{hb}(x) + l_{threshold}, \\ p_{hb}(y) = k_{hb}(y) + l_{threshold}. \end{cases} \\ p_{ht}(y) = k_{ht}(y) - l_{threshold}, \end{cases} \quad (8)$$

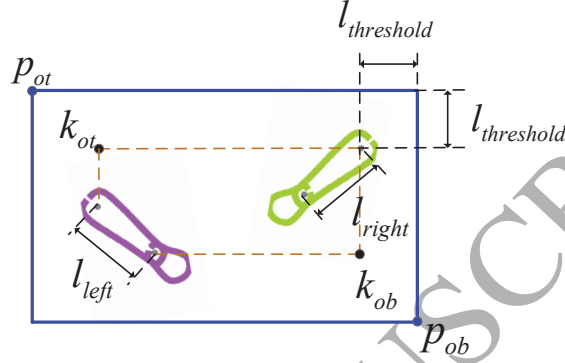


Figure 5: Calculation and selection of the operation area.

217 The calculation of operation patch is shown in Fig. 5. Operation patch represents the visual details
 218 around human actor's lower arms. We first obtain the top left k_{ot} and bottom right k_{ob} points of left/right
 219 elbows and left/right wrists. Then, we calculate the lengths l_{left} and l_{right} , which represent the length of
 220 left and right lower arms, respectively. Next, the top left (p_{ot}) and the bottom right (p_{ob}) endpoints of the
 221 operation area are computed as shown in Eq. (9).

$$\begin{cases} p_{ot}(x) = k_{ot}(x) - l_{threshold} - \frac{1}{2} \times l_{max}, & \begin{cases} p_{ob}(x) = k_{ob}(x) + l_{threshold} + \frac{1}{2} \times l_{max}, \\ p_{ob}(y) = k_{ob}(y) + l_{threshold} + \frac{1}{2} \times l_{max}, \end{cases} \\ p_{ot}(y) = k_{ot}(y) - l_{threshold} - \frac{1}{2} \times l_{max}, \end{cases} \quad (9)$$

222 where, $l_{max} = \max(l_{left}, l_{right})$ represents the maximum length of lower arms.

223 The operation area is the region from p_{ot} to p_{ob} . We resize this region to be 224×224 , and get the
 224 operation patch as shown in Fig. 6(c).

225 3.2.2. Motion cues

226 As well as obtaining appearance information from the extracted three image patches, we calculate optical
 227 flow images between adjacent video frames using the method proposed by Brox *et al.* [48], Fig. 7 shows the
 228 magnitude image of the two-dimension optical flow image for two adjacent images calculated using the

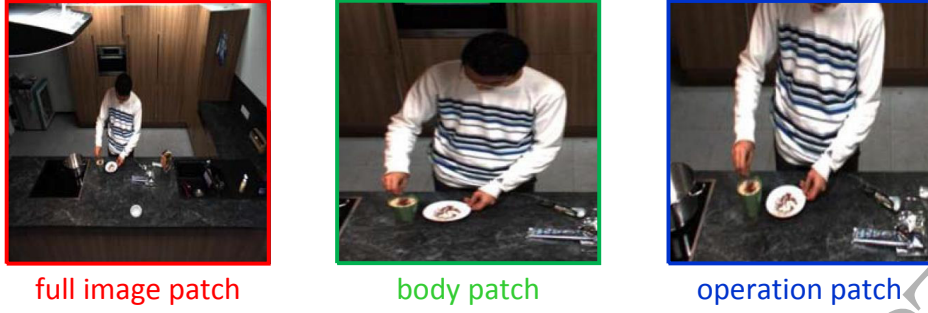


Figure 6: Three RGB patches for each frame of fine-grained action video sequence.

229 method of Brox *et al.* [48]. The optical flow image for each frame $t - 1$ and t is represented as \hat{U}_t :

$$\hat{U}_t = \begin{cases} f_{flow}(I_t, I_{t+1}), & \text{if } t = 1 \\ f_{flow}(I_{t-1}, I_t), & \text{otherwise} \end{cases} \quad (10)$$

230 where, $f_{flow}(\cdot)$ represents the method of calculating the optical flow matrix for two images proposed by [48].

231 According to the requirement of flow-CNN (described in Section 3.3), patches should be of three channels,
 232 so we need to transfer every $[a \times b \times 2]$ dimension optical flow image \hat{U}_t which is calculated by Eq. (10) into
 233 an $[a \times b \times 3]$ matrix. This has been performed in the following steps. First, using Eq. (11) we calculate
 234 $|U(x, y)|$, which is the magnitude of \hat{U}_t . Later, this magnitude is processed using the linear transformation
 235 shown in Eq. (12). The resultant matrix of this step serves as the additional third channel of the new optical
 236 flow image U_t . Next, the first two channels of U_t are calculated using Eq. (13). Finally, all three channels
 237 are concatenated to form the new optical flow image U_t . Fig. 8 shows the computed three channels of the
 238 optical flow image.

$$|U(x, y)| = \sqrt{\hat{U}_t(x, y, 1)^2 + \hat{U}_t(x, y, 2)^2} \quad (11)$$

$$U_t(x, y, 3) = 16 \times |U(x, y)| \quad (12)$$

$$U_t(x, y, c) = 16 \times \hat{U}_t(x, y, c) + 128, \quad \text{where } c = 1, 2 \quad (13)$$

239 Similar to the RGB patches in Fig. 6, we also extract three patches for each frame, *i.e.* full image patch,
 240 body patch and operation patch, as shown in Fig. 9.

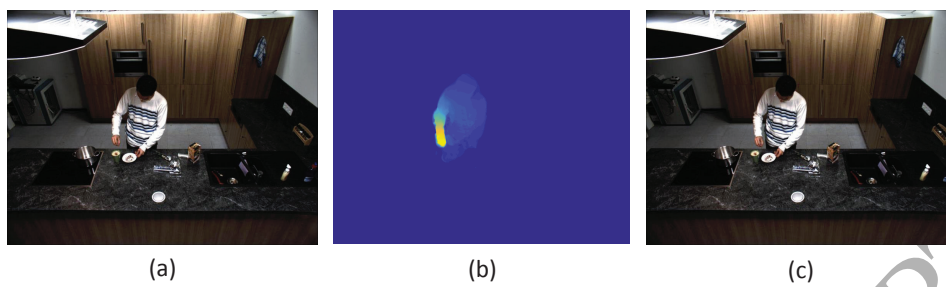


Figure 7: Calculation of the optical flow image for adjacent two video frames. (a) The image of frame $t - 1$; (b) The optical flow magnitude between the frame $t - 1$ and t ; and (c) The image of frame t .

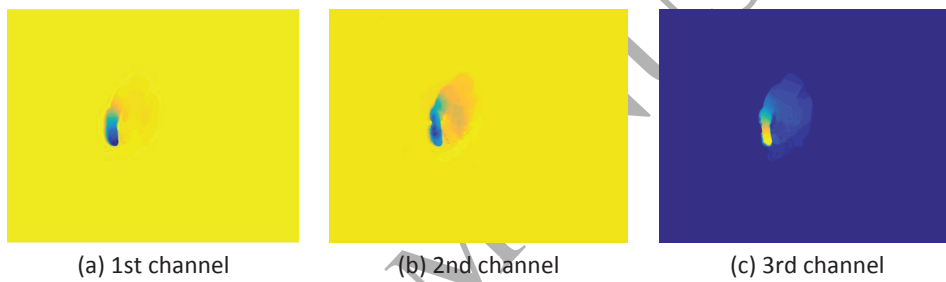


Figure 8: Three channels of optical flow patches for each frame of fine-grained action video sequence.



Figure 9: Three optical flow patches for each frame of fine-grained action video sequence.

241 3.3. Convolutional Neural Network Structures and Action Descriptor Construction

242 Considering that the properties for RGB images and flow images are different, we use similar but different
 243 CNN networks for RGB and flow patches separately. For simplicity, we call them RGB-CNN and flow-CNN
 244 respectively. Each of these networks have five convolutional layers and three pooling layers, as shown in
 245 Table 2. In Table 2, $n_1 \times k_1 \times k_1$ for convolutional layers mean using n_1 numbers of $k_1 \times k_1$ kernels, and
 246 $k_2 \times k_2$ for pooling layers mean using $k_2 \times k_2$ kernels. Besides, in Table 2, “str” means stride, and “pad” means
 247 padding. To make our proposed method more general, unlike some other related works which train data and
 248 parameters for each dataset specifically and individually, we use the same pre-trained CNN configurations
 249 for all the video sequences of various datasets. In our method, the data and parameters of the RGB-CNN
 250 architecture are pre-trained by Chatfield and Simonyan [49] using ILSVRC-2012 dataset [50]; while the data
 251 and parameters of the flow-CNN are trained by Gkioxari and Malik [42] using UCF101 dataset [51].

Table 2: CNN structures

Layer	$conv_1$	$pool_1$	$conv_2$	$pool_2$	$conv_3$	$conv_4$	$conv_5$	$pool_5$	f_{c_6}	f_{c_7}
RGB-CNN	64x11x11 str 4 pad 0	2x2	256x5x5 str 1 pad 2	2x2	256x3x3 str 1 pad 1	256x3x3 str 1 pad 1	256x3x3 str 1 pad 1	2x2	4096	4096
flow-CNN	96x7x7 str 2	3x3	384x5x5 str 2	3x3	512x3x3 str 1	512x3x3 str 1	384x3x3 str 1	3x3	4096	4096

252 As depicted in Table 2, $pool_5$ refers to the features of the last pooling layer, while f_{c_6} and f_{c_7} represent
 253 the features of first and second fully connected layers, respectively. Most current related works, for example,
 254 Girshick *et al.* [15] and Cheron *et al.* [10], use features of f_{c_6} or f_{c_7} to construct image descriptors, as
 255 illustrated in Fig. 10(a). However, other works claim that $pool_5$ contains more spatial information and
 256 could provide more distinguishable descriptors [52]. In our proposed work, we use the features of $pool_5$ to
 257 construct image descriptors, as illustrated in Fig. 10(b). Consider RGB-CNN for instance, the $pool_5$ layer
 258 has $6 \times 6 \times 256$ features, which corresponds to a vector of 9216 dimension, while f_{c_6} and f_{c_7} features are
 259 of 4096 dimension. In order to reduce the computational cost, we obtain reduced dimension by encoding.
 260 The most common used encoding methods are Fisher vector encoding [53] and vector of locally aggregated
 261 descriptors (VLAD) [54]. Xu *et al.* [52] analyzed the discriminating ability of Fisher and VLAD encoding
 262 methods, and found that VLAD is more feasible for CNN descriptors.

263 Assuming that the size of features in $pool_5$ layer is $a \times a \times b$, we reshape them into a^2 features, each of
 264 which is of b dimension, and we denote the new reshaped features as $\{x_1, x_2, \dots, x_{a^2}\}$. For entire video with T
 265 number of frames, a^2T features are obtained, which are denoted as $\{x_1, x_2, \dots, x_{T a^2}\}$. For each fine-grained
 266 action video, we extract six image patch sequences as shown in Fig. 6 and Fig. 9. We feed each image patch
 267 into corresponding CNN structure, and consequently we get a^2T features of b dimension for each patch
 268 sequence. Imaging that how huge amount the total features for fine-grained action sequence should be. In

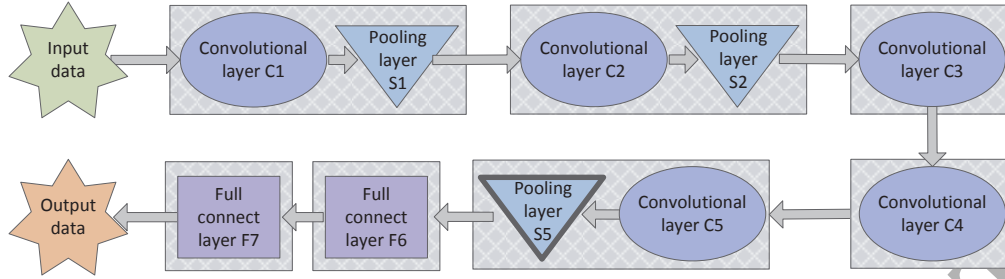
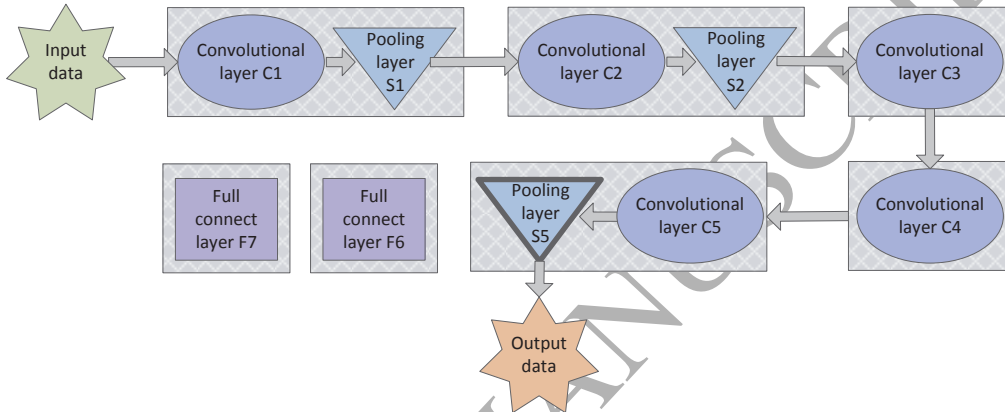
(a) Common usage of CNN architecture, which extract data from fc_7 layer.(b) Our proposed usage of CNN architecture, which extract data from $pool_5$ layer.

Figure 10: Emphasize CNN architecture used in our proposed system.

269 order to extract efficient information from these CNN features, we propose an encoding method. Note that
 270 different sequences focus on different information, for instance, RGB upper body patches focus on human
 271 pose spatial information, optical flow operation patches focus on hands and object motion information, etc.
 272 In this case, we handle each type of patch sequence separately. In the following description, we use RGB
 273 operation patch sequence as an example, but the handling method is the same for the other five sequence.

274 We assume that the number of fine-grained action video in the training dataset is N_{train} , and the length
 275 of all these videos are $\{T_1, T_2, \dots, T_{N_{train}}\}$ separately. For RGB operation patch sequence, the number of
 276 b dimension features are $(T_1 + T_2 + \dots + T_{N_{train}}) \times a^2$. We use k-means [55] to cluster these features, see
 277 Algorithm 1.

278 Based on Algorithm 1, we obtain K cluster centers for all RGB operation patch sequences of training
 279 dataset, denoted as $\{c_1, c_2, \dots, c_K\}$. Then for each sequence of testing dataset, we cluster the corresponding
 280 a^2T features of b dimension, and then calculate the nearest cluster center for each feature, and store the

Algorithm 1 Use k-means to cluster $pool_5$ features.

```

1: Select  $K = 128$  cluster centers of dimension  $b$ ;
2: while centers not stable do
3:   for  $i = 1, 2, \dots, (T_1 + T_2 + \dots + T_{N_{train}})$  do
4:     for  $j = 1, 2, \dots, K$  do
5:       Calculate distance between  $i$ th vector and  $j$ th center, denoted as  $d_{i,j}$ ;
6:     end for
7:     Calculate  $k_i = \underset{j=1,2,\dots,K}{\operatorname{argmin}} d_{i,j}$  as the index of cluster center for  $i$ th vector;
8:   end for
9:   for  $j = 1, 2, \dots, K$  do
10:    Calculate the mean location of vectors belong to the  $j$ th cluster as the updated  $j$ th center;
11:  end for
12: end while
13: Output the updated  $K$  centers.

```

281 index as $NN(x_i)$, as follows:

$$NN(x_i) = \underset{j=1,2,\dots,K}{\operatorname{argmin}} \|x_i - c_j\|. \quad (14)$$

282 Focusing on each cluster center c_j , we calculate distance between c_j and x_i which satisfies $NN(x_i) = j$,
283 and summarize the distance as u_j :

$$u_j = \sum_{x_i: NN(x_i)=j} (x_i - c_j). \quad (15)$$

284 Now, the VLAD encoding vector is obtained by concatenating all the u_j of k-means cluster centers, which
285 is denoted as $f = [u_1, u_2, \dots, u_K]$. The dimension of feature f is $b \cdot K$.

286 We use three RGB patches and three flow patches for each frame, and we suppose that the size of $pool_5$
287 features of the RGB-CNN and the flow-CNN are $a_1 \times a_1 \times b_1$ and $a_2 \times a_2 \times b_2$, respectively. We denote
288 VLAD descriptor of RGB full image patch, body patch, and operation patch as f_1 , f_2 , and f_3 separately;
289 we denote VLAD descriptor of optical flow image patch, body patch, and operation patch as f_4 , f_5 , and
290 f_6 respectively. Finally, we concatenate the obtained VLAD descriptors, to generate a fine-grained action
291 descriptor for video as follows:

$$f_{action} = [f_1, f_2, f_3, f_4, f_5, f_6]. \quad (16)$$

292 The obtained fine-grained action descriptor has $3(b_1 + b_2)K$ dimensions. In our implementation, we
293 calculate f_{action} of training dataset, and then use these video descriptors to train SVM (Support Vector
294 Machines) classifier [56]. The obtained SVM classifier is able to recognize fine-grained actions from different
295 videos.

296 4. Experiments

297 4.1. Dataset

298 Two publicly available human action datasets have been used for evaluation experiments. The first one
 299 is sub-JHMDB dataset [57], as shown in Fig. 11, which is proposed by Jhuang *et al.* contains 316 videos of
 300 twelve different actions, which are catch, climb stairs, golf, jump, kick ball, pick, pull-up, push, run, shoot
 301 ball, swing baseball and walk. This dataset is provided with three kinds of training/testing split modes. The
 302 other dataset is MPII Cooking dataset [7], as shown in Fig. 12, which contains 65 different cooking activities
 303 such as cut slices, pour spice, wash objects, etc., recorded from 12 participants. In total there are 44 videos
 304 with a total length of more than 8 hours. This dataset is provided with seven kinds of training/testing split
 305 modes. In our experiments, we test on each split mode of each dataset, and calculate the average accuracy
 306 among all splits.

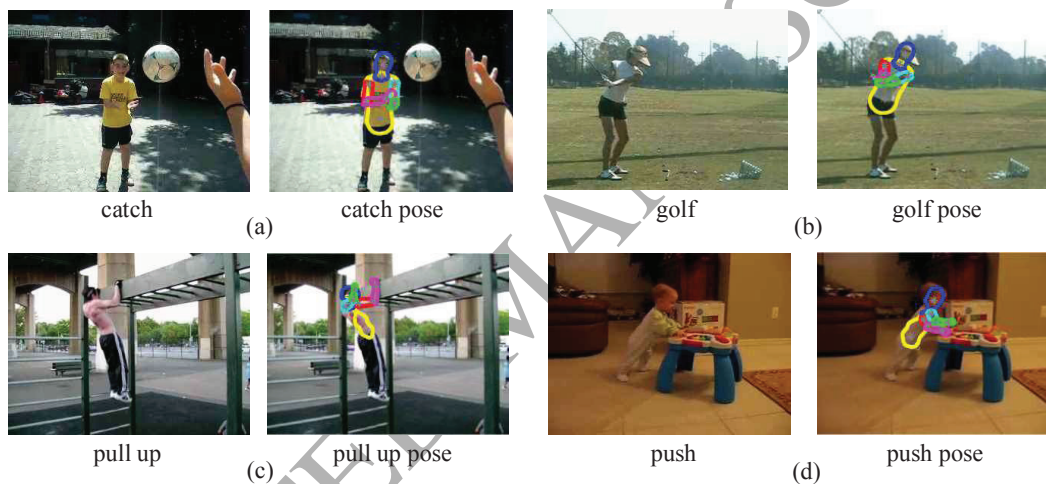


Figure 11: Sample frames of sub-JHMDB dataset. We show each frame twice: firstly the original video image; secondly the image overlaid with the estimated human pose which is used to extract human region patches.

307 From Fig. 12 it is seen that with MPII Cooking dataset, most human action information is contained
 308 in the upper body, while with sub-JHMDB dataset (see Fig. 11), the action information is encoded in the
 309 entire human body pose. In this case, for the computation of body path on the sub-JHMDB dataset, we
 310 obtain the person's torso bottom position k_{belly} estimated in Section 3.1, and mirror the human upper body
 311 (obtained in Section 3.2) vertically using the center of k_{belly} , as shown in Fig. 13(b). The region within the
 312 upper body rectangle patch and the vertically mirrored patch is denoted as body patch for the calculation
 313 on the sub-JHMDB dataset, as shown in Fig. 13(c).

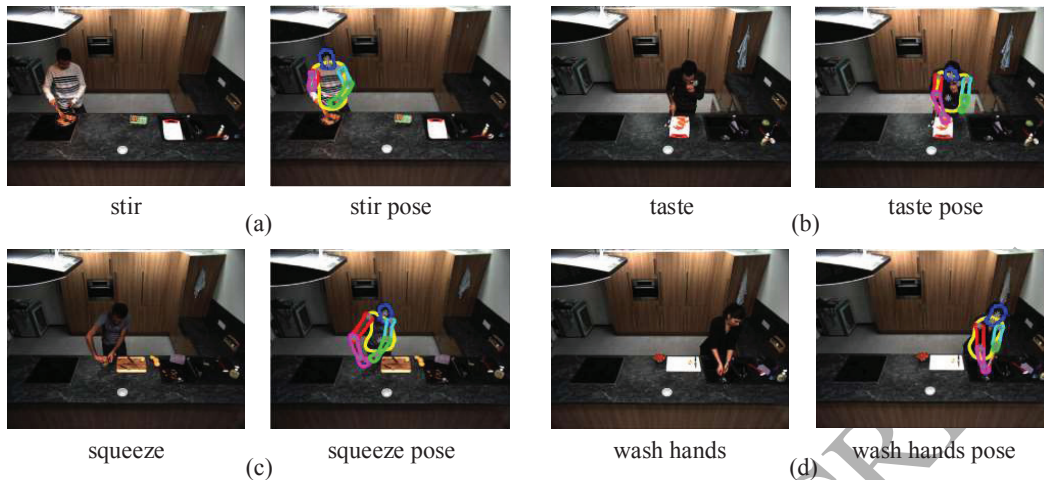


Figure 12: Sample frames of MPII Cooking dataset. We show each frame twice, the first one is the original video frame from dataset, the second one is overlaid with the estimated human pose which is used to extract human region patches.

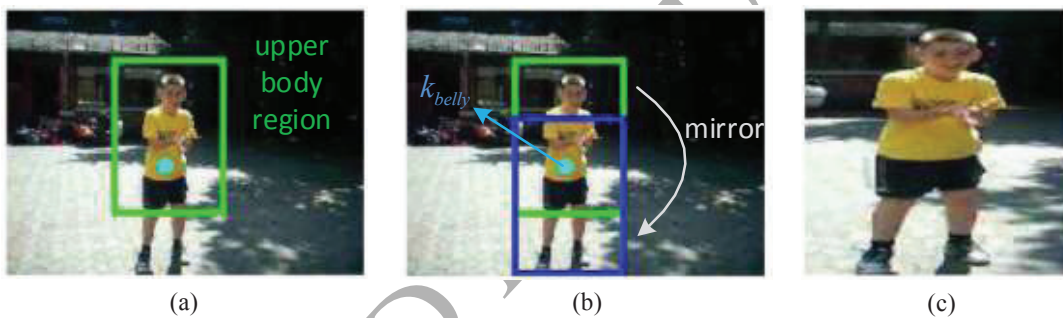


Figure 13: Obtaining body patches for the sub-JHMDB dataset. (a) Human upper body patch (green); (b) Mirrored green patch vertically with the center of belly button to get blue patch; (c) The obtained whole body patch.

314 4.2. Preliminaries

315 In video sequences, human movement is typically continuous. However, through observation we found that
 316 some adjacent frames are similar while others have significantly different human position and pose. That is
 317 to say, different frames contribute different amounts of information for human fine-grained action recognition
 318 in videos. In this case, the calculation of similar frames in video may lead to redundant information and
 319 increase the amount of calculation.

320 The problem described above is not serious in short sequences. However, the calculation of k-means
 321 centers for long videos of the MPII Cooking dataset as expounded in Algorithm 1 causes very large cost
 322 of computation time and computation space. To solve this problem, when calculating k-means centers for
 323 $pool_5$ layer (Algorithm 1), we calculate and select key frames of each video sequence instead of using all the

324 frames. This has been accomplished as follows and illustrated in Fig. 14.

325 For every video (whose length is denoted to be T), we use the features of f_{c7} layer from the CNN structure
 326 of the RGB full image patches to form a $T \times 4096$ matrix. The columns whose elements are all zeros are
 327 removed, and then we select the largest elements (denoted as “m”) from the remaining matrix (denoted as
 328 C). We transfer every element of C into binary sequence with the length of $\log_2 m$. In this case, the obtained
 329 new matrix is denoted by \hat{C} , as shown in Fig. 14.

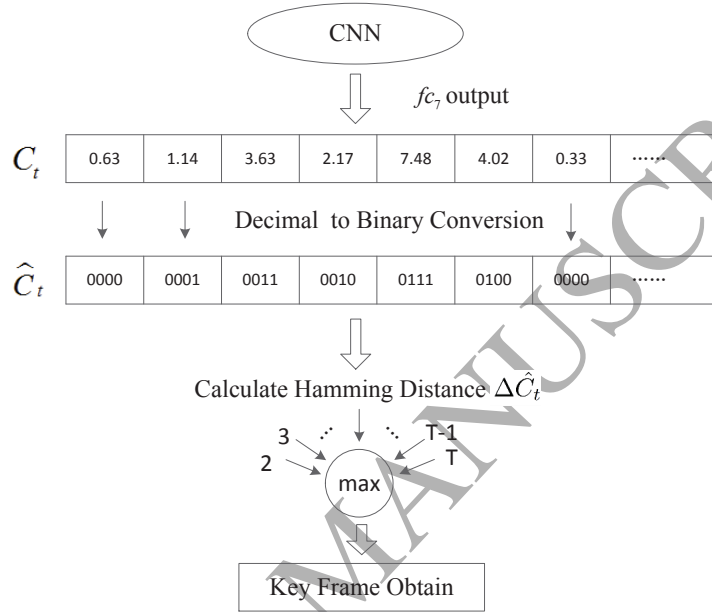


Figure 14: Calculate and select key frames from video sequence using CNN f_{c7} layer output.

330 For $t = 2, 3, \dots, T$, we calculate the *hamming distance* between \hat{C}_{t-1} and \hat{C}_t , which is denoted as $\Delta \hat{C}_t$,
 331 see Eq. (17).

$$\Delta \hat{C}_t = \text{hamming}(\hat{C}_t - \hat{C}_{t-1}), \quad t = 2, 3, \dots, T \quad (17)$$

332 Ranking all the frames according to descending order of the hamming distance $\Delta \hat{C}_t$, we select k frames with
 333 the largest hamming distance, which are called key frames. These key frames are then used to calculate the
 334 k-means centers for the VLAD encoding.

335 The number of k-means cluster centers K (see Algorithm 1) should always be greater than the total
 336 number of actions. This can be justified by analyzing the clustering performance with different values of
 337 K . For instance, consider the calculation of cluster centers (as described in Algorithm 1) for the RGB full
 338 image patch sequence of the MPII Cooking dataset, which has 65 actions. For a range of $K = [32, 256]$,
 339 the average value of cluster radiuses (average cluster radius, ACR) are calculated. A cluster radius is the
 340 largest distance between each point and the corresponding cluster center, and can be used for evaluating the

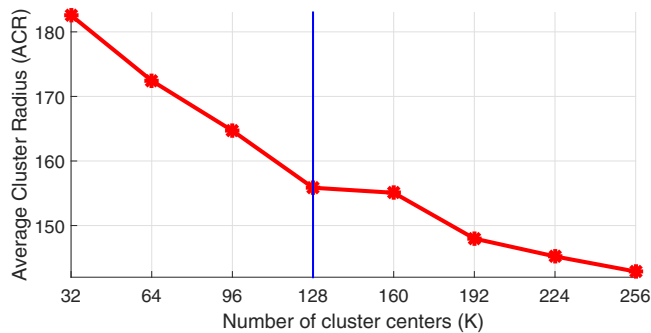


Figure 15: Relationship between the number of cluster centers K and the average cluster radius. Smaller the ACR better the clustering performance.

clustering performance *i.e.*, smaller the ACR better the performance. From Fig. 15, it can be seen that the performance improves with the number of cluster centers (larger K). Although the performance is improved, with higher values of K , overall computation time also increases. Hence, an appropriate balance between cluster performance and calculation time should be maintained while selecting K . From Fig. 15 we can see that the performance rapidly improved until $K = 128$ and progressed gradually afterwards. Therefore, in our experiments, we choose the number of cluster centers to be $K = 128$ (step 1 of Algorithm 1).

Also it is worth mentioning that all the experiments are conducted on a lab computer running Ubuntu 14.04 with 2.80GHz Intel Core i7 CPU and 16 GB of RAM. We have used Matlab for implementation purposes.

4.3. Action classification performance

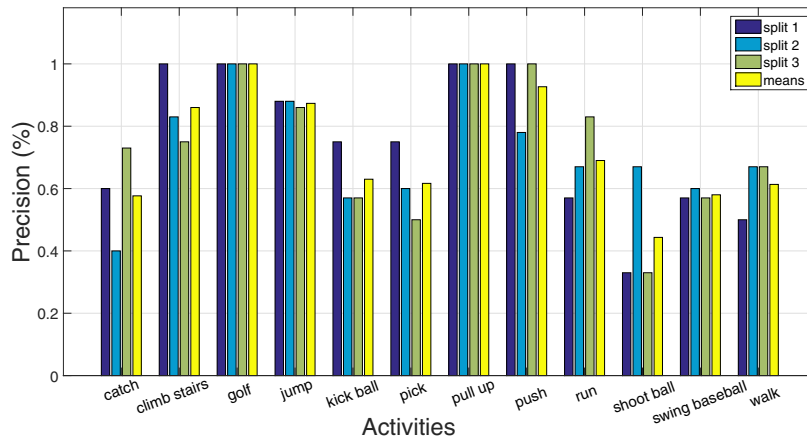
Our proposed action recognition system is able to classify fine-grained actions, but also works well for general physical actions. For the reason to verify that our proposed action recognition method is not limited to fine-grained actions, we use a very commonly used public human action dataset to test the system's performance. In this section, we use sub-JHMDB dataset to test the precision and recall performance of our proposed system. The definition of evaluation criterion "precision" and "recall" are described in Eq. (18) and Eq. (19) separately.

$$precision = \frac{count((recognized\ as\ action\ A) \cup (action\ A\ in\ dataset))}{count(recognized\ as\ action\ A)} \quad (18)$$

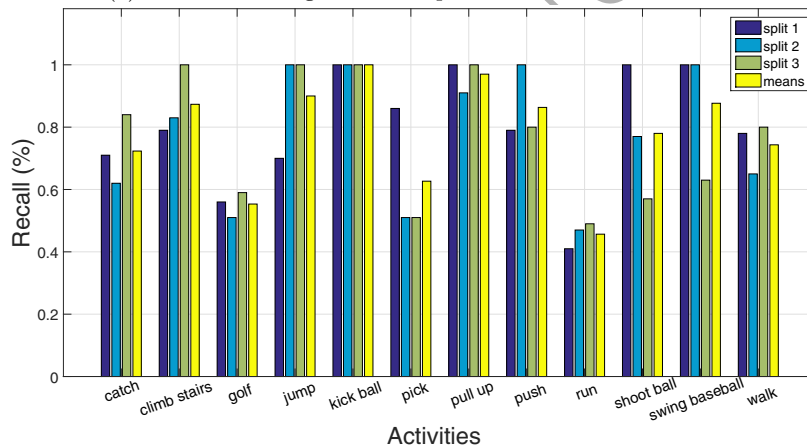
$$recall = \frac{count((recognized\ as\ action\ A) \cup (action\ A\ in\ dataset))}{count(action\ A\ in\ dataset)} \quad (19)$$

The detailed per-class human actions classification results are shown in Fig. 16. From Fig. 16(a) we can see that the action classification precision of actions such as golf, pull up, push, jump and climb stairs are very

high, with mean precision of 100%, 100%, 92% 87% and 86% respectively. The mean action classification
 precision of catch, kick ball, pick, run, swing baseball and walk are 57%, 63%, 61%, 69%, 58%, 61% and 73%
 respectively, which is somewhat less good, but is still highly competitive compared with other state-of-the-art
 methods. The mean precision of shoot ball is the lowest, at 44%.



(a) Precision bar figure for all splits of sub-JHMDB dataset.



(b) Recall bar figure for all splits of sub-JHMDB dataset.

Figure 16: Performance on the dataset of sub-JHMDB.

Fig. 16(b) reveals the recall of each training/testing split for sub-JHMDB dataset. As presented in
 Fig. 16(b), the mean recall of actions such as climb stairs, jump, kick ball, pull up, push and swing baseball
 are more than 80%, while catch, pick, shoot ball and walk are between 60% - 80%. The mean recall of golf
 and run are 55% and 46% respectively. From the performance revealed in Fig. 16, it is obvious that our
 proposed method performs well for common physical action recognition problem.

368 4.4. Comparison with state-of-the-art methods

369 In this section, we present the experimental results obtained by validating our proposed human actions
 370 classification method against various other state-of-the-art methods proposed by Wang *et al.* [26], Gkioxari
 371 *et al.* [42], Peng *et al.* [24] and Cheron *et al.* [10] on the sub-JHMDB dataset (Table 3), and against the
 372 methods of Ni *et al.* [58] and Rohrbach *et al.* [7] on MPII Cooking dataset (Table 4). The reason that we
 373 use different comparison methods for different testing dataset is that the compared methods are the state-of-
 374 the-art methods which have been published on each dataset respectively. I.e. we compare against the best
 375 methods on each dataset for which performance data is publicly available.

Table 3: Performance on sub-JHMDB dataset compared with related state-of-the-art methods.

Method	Accuracy
Wang <i>et al.</i> [26]	56.6%
Gkioxari <i>et al.</i> [42]	62.5%
Peng <i>et al.</i> [24]	69.3%
Cheron <i>et al.</i> [10]	72.2%
Ours	76.9%

376 Table 3 shows our proposed system performance on sub-JHMDB dataset compared with related state-
 377 of-the-art methods. The “accuracy” in Table 3 means the proportion of correct recognized in all the videos
 378 of test dataset. As reported by Wang *et al.* [26], they proposed a method which describe videos by sampling
 379 dense points from each frame and tracking them based on displacement information from a dense optical
 380 flow field. Gkioxari and Malik [42] proposed a method which selects image regions containing salient motion
 381 and use spatial and temporal information to build action representations. These two methods extract image
 382 regions utilizing motion information, while we use our human pose tracking method to obtain consecutive
 383 regions which contain human action information. Table 3 reveals that on the sub-JHMDB dataset the
 384 accuracy of our proposed method is **20.3%** higher than the method of Wang *et al.* [26], and **14.4%** higher
 385 than the method of Gkioxari and Malik [42].

386 Peng *et al.* [24] proposed *stacked Fisher vectors* to represent action features. This method uses Fisher
 387 vectors to encode local features extracted from the densely sampled sub-volumes in the first layer, and
 388 compress the sub-volumes Fisher vectors as well as encodes them again with Fisher vectors. Cheron *et*
 389 *al.* [10] proposed a method which aggregates motion and appearance information along tracks of human
 390 body parts through CNN structures. Instead of using stacked Fisher vectors or soft-max CNN features, we
 391 proposed a VLAD encoding method using *pool₅* layer features. From Table 3 we can see that the accuracy
 392 of our proposed method is **7.6%** higher than the method by Peng *et al.* [24], and **4.7%** higher than the
 393 method of Cheron *et al.* [10].

394 Table 4 shows our proposed system performance on MPII Cooking dataset compared with related state-

Table 4: Performance on MPII Cooking dataset compared with related state-of-the-art methods.

Method	Precision	Recall	Average Precision (AP)
Ni <i>et al.</i> [58]	28.6%	48.2%	54.3%
Rohrbach <i>et al.</i> [7]	50.4%	45.1%	57.9%
Ours	57.7%	56.4%	70.3%

of-the-art methods. The “Average Precision (AP)” in Table 4 means the performance obtained by drawing a precision-recall curve based on the SVM classification score and calculating the area under the curve (AUC).

Ni *et al.* [58] proposed a strategy to detect and define the interaction between hands and objects. This strategy infers coarse interaction status and uses the obtained information to get compact action feature. Instead of focusing only on hand and object to construct action features, our proposed method extracts image patches according to the estimated human pose and uses six different patch sequences containing spatial and temporal information. From Table 4 it can be seen that the precision and recall of our proposed method is **29.1%** and **8.2%** higher than the method of Ni *et al.* , and the average precision of our method is **16%** higher than Ni *et al.* method.

Rohrbach *et al.* [7] proposed a method which estimates human pose joint points among frame sequence, and compute a separate codebook for each distinct sub-feature (*i.e.* velocity, acceleration, exponential bands etc.), then extracts histograms of oriented gradients (HOG), histograms of optical flow (HOF), and motion boundary histograms (MBH) around densely sampled points. With all these features and methods, the average precision of [7] is 57.9%, and their precision and recall are 50.4% and 55.1% respectively. Instead of relying on hand-crafted features, we use the $pool_5$ layer features of CNN structure, which are more intelligent and distinguishable. The results are summarized in Table 4. It can be seen that our precision, recall and average precision are **7.3%**, **11.3%** and **12.4%** higher than Rohrbach *et al.* .

4.5. Contribution of each system component to the overall performance

In this section we examine the contribution of each part of our proposed method to overall performance on MPII Cooking dataset. With our method, firstly we proposed a pose tracking and estimation schema for constructing human action region patches. To verify the efficiency of pose estimation method, we compute the CNN features based only on RGB and flow full-image patches. The obtained results are shown in the **first row** of Table 5. From these, we notice that the human action recognition precision and recall, without using our proposed body patches and operation patches approach, are 37.3% and 43.5%, which are lower than our full method by 20% and 12.9% respectively. The average precision by using only full image patch is 12.9% lower than our patch-based method. This is because the background of full images contains large amounts of redundant information that is less useful than foreground patches. In contrast, our human pose estimation method is able to extract and enhance foreground patches and make better use of the most

Table 5: Contribution of each system component to the overall performance on MPII Cooking dataset.

No.	Method	Precision	Recall	Average Precision (AP)
1	fp(both)+ <i>pool</i> ₅ +VLAD	37.7%	43.5%	57.4%
2	fp(both)+bp(both)+ <i>pool</i> ₅ +VLAD	41.8%	50.9%	64.6%
3	fp(RGB)+bp(RGB)+op(RGB)+ <i>pool</i> ₅ +VLAD	48.4%	46.5%	67.8%
4	fp(both)+bp(both)+op(both)+ <i>fc</i> ₇ +max	42.1%	53.0%	62.1%
5	fp(both)+bp(both)+op(both)+ <i>fc</i> ₇ +average	30.8%	28.7%	36.4%
6	fp(both)+bp(both)+op(both)+ <i>fc</i> ₇ +VLAD	47.1%	54.9%	64.2%
7	fp(both)+bp(both)+op(both)+<i>conv</i>₄+VLAD	36.2%	42.3%	40.7%
8	fp(both)+bp(both)+op(both)+ <i>pool</i> ₅ +VLAD	57.7%	56.4%	70.3%

¹ “fp” means full image patch, “bp” means body patch, and “op” means operation patch.

² In the parentheses: “RGB” means using RGB patches (see Fig. 6), while “both” means using both RGB patches and optical flow patches (see Fig. 9).

423 information-rich parts of each image.

424 An additional contribution of our work is to define and extract the operation areas for human actions,
 425 which are denoted by operation patches as shown in Fig. 3. To verify the contribution of our operation
 426 patches to overall system performance, we test the MPII Cooking dataset with and without using our
 427 operation patches method. The results are shown in the **second row** of Table 5. These results reveal that,
 428 by using our operation patch method, the precision, recall and average precision are increased by 15.9%,
 429 5.5% and 5.7% respectively. The reason is that the operation patch emphasizes the information around hand
 430 regions which includes hand pose, object, as well as hands interaction information. Human fine-grained
 431 action information mostly exists around lower arms, while common action recognition methods often resize
 432 the original images and indirectly reduce effective pixels. Our proposed method overcomes this problem
 433 by extracting human operation areas and enhancing the corresponding pixels. This improves the action
 434 recognition results.

435 As mentioned before, RGB images offer appearance cues for human actions, while the calculated optical
 436 flow image sequence contains motion cues. In order to verify that the motion information from the optical flow
 437 images helps to get better results in recognizing human actions, we test our method with and without optical
 438 flow images separately. The results are summarized in the **third row** of Table 5. The results illustrate that
 439 without the motion information obtained from optical flow images, precision, recall and average precision
 440 would decline by 9.3%, 9.9% and 2.5% respectively (compared with the eighth row of Table 5). The reason for
 441 the better performance is that the temporal information contained in optical flow images offers human action
 442 motion cues, which contributes to human action recognition problem and improves the system performance.

443 We use *pool*₅ features with VLAD encoding method instead of the common handling of *fc*₇ features. In
 444 order to justify this, we compare our results with the two most commonly used handling methods for *fc*₇
 445 features. Specifically, we connect the *fc*₇ features of every image patch per frame as frame descriptor, and
 446 calculate the maximum and average number of each dimension of frame descriptors per video separately

447 as video descriptors. The obtained video descriptors are then used to train the SVM classifier in order to
 448 generate the human action recognition results. The results are shown in the fourth and fifth rows of Table 5.
 449 It can be seen from the **fourth row** that for the fc_7 +max method, the results are lower by 15.6%, 3.4%
 450 and 8.2% on precision, recall and average precision, respectively than our $pool_5$ +VLAD method (the eighth
 451 row in Table 5). The reason is that the “max” calculation would get rid of some useful information and only
 452 keep the extreme values caused by distractions. In this case, the remained extreme value features would not
 453 be bale to represent the action properly and lead to failure, especially in the case of strong light or other
 454 object distractions.

455 The **fifth row** of Table 5 illustrates the results of using fc_7 +average method. The results in this case
 456 are much lower than the previous, *i.e.*, lower by 26.9%, 27.7% and 33.9% on precision, recall and average
 457 precision respectively, compared with the proposed $pool_5$ +VLAD method. The reason is that the “average”
 458 calculation only keeps the most common and average values that are not discriminative. However, different
 459 fine-grained actions in MPII Cooking dataset are quite similar and the distinguishable features lie only in
 460 some specific frames or locations. As a result, the “average” calculation would make the descriptors of
 461 different fine-grained actions similar and lose the discrimination quality, due to which the results become
 462 lower. However, the proposed VLAD encoding method (see the eighth row in Table 5) does not have this
 463 problem and is able to organize video descriptors more effectively, even though these features are of less
 464 dimension.

465 Furthermore, to demonstrate that $pool_5$ layer output data contains more action information than fc_7
 466 layer, *i.e.*, spatial information, we test our method employing fc_7 layer outputs instead of $pool_5$ layer outputs,
 467 leaving the other methods and experimental settings (*e.g.* parameters) unchanged. Specifically, we calculate
 468 the k-means centers of 4096 dimensions for fc_7 layer outputs on the training dataset, and use VLAD encoding
 469 method to encode fc_7 layer outputs of all the frames for each video sequence into video descriptors. The
 470 results are shown in the **sixth row** of Table 5. By comparing these results with our $pool_5$ +VLAD (results in
 471 eighth row), it is illustrated that using the outputs of $pool_5$ layer, the precision, recall and average precision
 472 are 10.6%, 1.5% and 6.1% higher than that of using the outputs of fc_7 layer. The main reason is that $pool_5$
 473 layer features contain more abundant spatial information than fc_7 layer features, so the corresponding data
 474 are more effective. From these results, it is clear that the proposed feature handling method achieves higher
 475 performance in recognizing fine-grained actions in video sequences.

476 Moreover, in CNN structures, the aim of convolutional layers is to extract certain features such as edges,
 477 angles, curves, or more complex higher-order features. Even though the outputs of earlier layers contain
 478 more spatial information, the features are eminently raw and are less distinguishable. In order to justify that
 479 using the $pool_5$ layer features make an appropriate balance between distinguishable features and adequate
 480 spatial information, we test our method employing the fourth convolutional layer $conv_4$ outputs with VLAD

481 encoding (see Table 2). The data size of the $conv_4$ layer outputs is similar to that of the $pool_5$ layer (which
482 can be recorded as $a \times a \times b$), therefore the processing procedure is the same. The corresponding experimental
483 results are shown in the **seventh row** of Table 5. By comparing the seventh and eighth rows of Table 5, we
484 found that using $conv_4$ layer data makes the precision, recall and average precision decline by 21.5%, 14.1%
485 and 29.6% respectively than that of using $pool_5$ layer features. Despite of the fact that $conv_4$ layer outputs
486 contain more spatial information, the contained features are hardly distinguishable. These results clearly
487 demonstrates that utilizing $pool_5$ layer outputs makes an appropriate balance between adequate spatial
488 information and distinguishable features, which once again proves the effectiveness of the proposed method
489 in recognizing human actions in videos.

490 5. Conclusion

491 We have proposed a novel region sequence based six-stream CNN feature for human action recognition in
492 videos, which combines different scales of image appearance information and video motion information. We
493 built on our recent state-of-the-art method for human pose estimation in video sequences, which localizes the
494 human body parts. As a result, we can then crop different scale human region image patches. This approach
495 uses human body part positions as prior knowledge, and makes better use of spatial image information, which
496 enables fine-grained activities to be distinguished. A variety of different scales of appearance and motion
497 patch sequences are processed in CNN structures to provides richer action information. The encoding of
498 the outputs of the CNN pooling layer gives more effective descriptors. Our method offers a general human
499 action recognition method for videos. Our proposed method outperforms six other state-of-the-art methods,
500 in empirical experiments on two publicly available challenging human action datasets.

501 References

- 502 [1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The pascal visual object classes
503 (voc) challenge, *International Journal of Computer Vision* 88 (2) (2010) 303–338.
- 504 [2] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in:
505 *Advances in Neural Information Processing Systems*, 2014, pp. 568–576.
- 506 [3] Y. Yang, Y. Li, C. Fermuller, Y. Aloimonos, Robot learning manipulation action plans by” watching”
507 unconstrained videos from the world wide web, in: *Advancement of Artificial Intelligence*, 2015, pp.
508 3686–3693.
- 509 [4] M. Niitsumag, H. Hashimoto, H. Hashimoto, Spatial memory as an aid system for human activity in
510 intelligent space, *IEEE Transactions on Industrial Electronics* 54 (2) (2007) 1122–1131.

- 511 [5] Z. A. Khan, W. Sohn, Abnormal human activity recognition system based on r-transform and kernel
512 discriminant technique for elderly home care, *IEEE Transactions on Consumer Electronics* 57 (4) (2011)
513 1843–1850.
- 514 [6] V. Krüger, D. Kragic, A. Ude, C. Geib, The meaning of action: a review on action recognition and
515 mapping, *Advanced Robotics* 21 (13) (2007) 1473–1501.
- 516 [7] M. Rohrbach, S. Amin, M. Andriluka, B. Schiele, A database for fine grained activity detection of
517 cooking activities, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012,
518 pp. 1194–1201.
- 519 [8] I. Laptev, M. Marszalek, C. Schmid, B. Rozenfeld, Learning realistic human actions from movies, in:
520 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.
- 521 [9] H. Jhuang, T. Serre, L. Wolf, T. Poggio, A biologically inspired system for action recognition, in: *IEEE*
522 *International Conference on Computer Vision (ICCV)*, 2007, pp. 1–8.
- 523 [10] G. Cheron, I. Laptev, C. Schmid, P-cnn: pose-based cnn features for action recognition, in: *IEEE*
524 *International Conference on Computer Vision (ICCV)*, 2015, pp. 3218–3226.
- 525 [11] B. Yao, S. Zhu, Learning deformable action templates from cluttered videos, in: *IEEE International*
526 *Conference on Computer Vision (ICCV)*, 2009, pp. 1507–1514.
- 527 [12] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei, Large-scale video classification
528 with convolutional neural networks, in: *IEEE Conference on Computer Vision and Pattern Recognition*
529 *(CVPR)*, 2014, pp. 1725–1732.
- 530 [13] C. Schüldt, I. Laptev, B. Caputo, Recognizing human actions: a local svm approach, in: *IEEE 17th*
531 *International Conference on Pattern Recognition*, Vol. 3, 2004, pp. 32–36.
- 532 [14] J. Liu, M. S. Jiebo Luo, Recognizing realistic actions from videos in the wild, in: *IEEE Conference on*
533 *Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 1996–2003.
- 534 [15] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and
535 semantic segmentation, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*,
536 2014, pp. 580–587.
- 537 [16] G. Gkioxari, R. Girshick, J. Malik, Contextual action recognition with r*cnn, in: *IEEE International*
538 *Conference on Computer Vision (ICCV)*, 2015, pp. 1080–1088.
- 539 [17] M. Ma, N. Marturi, Y. Li, R. Stolkin, A. Leonardis, A local-global coupled-layer puppet model for
540 robust online human pose tracking, *Computer Vision and Image Understanding* 153 (2016) 163–178.

- 541 [18] R. Poppe, A survey on vision-based human action recognition, *Image and Vision Computing* 28 (6)
542 (2010) 976–990.
- 543 [19] D. M. Gavrila, The visual analysis of human movement: A survey, *Computer Vision and Image Under-*
544 *standing* 73 (1) (1999) 82–98.
- 545 [20] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer*
546 *Vision* 60 (2) (2004) 91–110.
- 547 [21] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: *IEEE Conference on*
548 *Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 886–893.
- 549 [22] R. Chaudhry, Avinash, Ravichandran, G. Hager, R. Vidal, Histograms of oriented optical flow and
550 binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions, in: *IEEE*
551 *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 1932–1939.
- 552 [23] H. Wang, C. Schmid, Action recognition with improved trajectories, in: *IEEE International Conference*
553 *on Computer Vision (ICCV)*, 2013, pp. 3551–3558.
- 554 [24] X. Peng, C. Zou, Y. Qiao, Q. Peng, Action recognition with stacked fisher vectors, in: *European*
555 *Conference on Computer Vision (ECCV)*, 2014, pp. 581–595.
- 556 [25] A. Iosifidis, A. Tefas, I. Pitas, Discriminant bag of words based representation for human action recog-
557 nition, *Pattern Recognition Letters* 49 (2014) 185–192.
- 558 [26] H. Wang, A. Klaser, C. Schmid, C. L. Liu, Action recognition by dense trajectories, in: *IEEE Conference*
559 *on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3169–3176.
- 560 [27] H. Wang, A. Klaser, C. Schmid, C. L. Liu, Dense trajectories and motion boundary descriptors for
561 action recognition, *International Journal of Computer Vision* 103 (1) (2013) 60–79.
- 562 [28] C. Xu, D. Tao, C. Xu, A survey on multi-view learning, arXiv preprint arXiv:1304.5634.
- 563 [29] C. Xu, D. Tao, C. Xu, Multi-view learning with incomplete views, *IEEE Transactions on Image Pro-*
564 *cessing* 24 (12) (2015) 5812–5825.
- 565 [30] J. Li, C. Xu, W. Yang, C. Sun, D. Tao, Discriminative multi-view interactive image re-ranking, *IEEE*
566 *Transactions on Image Processing* 26 (7) (2017) 3113–3127. doi:10.1109/TIP.2017.2651379.
- 567 [31] M. Bregonzio, T. Xiang, S. Gong, Fusing appearance and distribution information of interest points for
568 action recognition, *Pattern Recognition* 45 (3) (2012) 1220–1234.

- 569 [32] J. C. Niebles, L. Fei-Fei, A hierarchical model of shape and appearance for human action classification,
570 in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2007, pp. 1–8.
- 571 [33] S. Ma, J. Zhang, N. Ikidler-Cinbis, S. Sclaroff, Action recognition and localization by hierarchical space-
572 time segments, in: IEEE International Conference on Computer Vision (ICCV), 2013, pp. 2744–2751.
- 573 [34] T. Lan, Y. Zhu, A. R. Zamir, S. Savarese, Action recognition by hierarchical mid-level action elements,
574 in: IEEE International Conference on Computer Vision (ICCV), 2015, pp. 4552–4560.
- 575 [35] P. Arbelaez, M. Maire, C. Fowlkes, J. Malik, From contours to regions: An empirical evaluation, in:
576 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 2294–2301.
- 577 [36] L. Liu, L. Shao, P. Rockett, Boosted key-frame selection and correlated pyramidal motion-feature rep-
578 resentation for human action recognition, *Pattern Recognition* 46 (7) (2013) 1810–1818.
- 579 [37] D. H. Hubel, T. N. Wiesel, Receptive fields, binocular interaction and functional architecture in the
580 cat’s visual cortex, *The Journal of Physiology* 160 (1) (1962) 106–154.
- 581 [38] K. Fukushima, Neocognitron: A self-organizing neural network model for a mechanism of pattern
582 recognition unaffected by shift in position, *Biological Cybernetics* 36 (4) (1980) 193–202.
- 583 [39] J. Wu, C. Leng, Y. Wang, Q. Hu, J. Cheng, Quantized convolutional neural networks for mobile devices,
584 in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4820–4828.
- 585 [40] E. P. Ijjina, K. M. Chalavadi, Human action recognition using genetic algorithms and convolutional
586 neural networks, *Pattern Recognition* 59 (2016) 199–212.
- 587 [41] Y. Wang, C. Xu, S. You, D. Tao, C. Xu, Cnnpack: packing convolutional neural networks in the
588 frequency domain, in: *Advances in Neural Information Processing Systems*, 2016, pp. 253–261.
- 589 [42] G. Gkioxari, J. Malik, Finding action tubes, in: IEEE Conference on Computer Vision and Pattern
590 Recognition (CVPR), 2015, pp. 759–768.
- 591 [43] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural net-
592 works, in: *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.
- 593 [44] T. Y. Lin, A. Roychowdhury, S. Maji, Bilinear cnn models for fine-grained visual recognition, in: IEEE
594 Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1449–1457.
- 595 [45] S. Zuffi, O. Freifeld, M. J. Black, From pictorial structures to deformable structures, in: IEEE Conference
596 on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3546–3553.

- 597 [46] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, J. Davis, Scape: shape completion and
598 animation of people, in: *ACM Transactions on Graphics (TOG)*, Vol. 24, 2005, pp. 408–416.
- 599 [47] S. Zuffi, J. Romero, C. Schmid, M. J. Black, Estimating human pose with flowing puppets, in: *IEEE*
600 *International Conference on Computer Vision (ICCV)*, IEEE, 2013, pp. 3312–3319.
- 601 [48] T. Brox, A. Bruhn, N. Papenberg, J. Weickert, High accuracy optical flow estimation based on a theory
602 for warping, in: *European Conference on Computer Vision (ECCV)*, 2004, pp. 25–36.
- 603 [49] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: Delving deep
604 into convolutional nets, *arXiv preprint arXiv:1405.3531*.
- 605 [50] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image
606 database, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 248–
607 255.
- 608 [51] K. Soomro, A. R. Zamir, M. Shah, Ucf101: A dataset of 101 human actions classes from videos in the
609 wild, *arXiv preprint arXiv:1212.0402*.
- 610 [52] Z. Xu, Y. Yang, A. G. Hauptmann, A discriminative cnn video representation for event detection, in:
611 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1798–1807.
- 612 [53] J. Sánchez, F. Perronnin, T. Mensink, J. Verbeek, Image classification with the fisher vector: Theory
613 and practice, *International Journal of Computer Vision* 105 (3) (2013) 222–245.
- 614 [54] H. Jégou, M. Douze, C. Schmid, P. Pérez, Aggregating local descriptors into a compact image rep-
615 resentation, in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp.
616 3304–3311.
- 617 [55] A. Vedaldi, B. Fulkerson, Vfeat: An open and portable library of computer vision algorithms, in:
618 *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 1469–1472.
- 619 [56] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, *ACM Transactions on Intelligent*
620 *Systems and Technology (TIST)* 2 (3) (2011) 27.
- 621 [57] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, M. J. Black, Towards understanding action recognition, in:
622 *IEEE International Conference on Computer Vision (ICCV)*, 2013, pp. 3192–3199.
- 623 [58] B. Ni, V. R. Paramathayalan, P. Moulin, Multiple granularity analysis for fine-grained action detection,
624 in: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 756–763.