




Article

Fine-Grained Action Recognition by Motion Saliency and Mid-Level Patches

Fang Liu ^{1,2}, Liang Zhao ^{2,*}, Xiaochun Cheng ³, Qin Dai ^{4,*} and Xiangbin Shi ²
and Jianzhong Qiao ¹

¹ School of Computer Science and Engineering, Northeastern University, Shenyang 110819, China; liufang@sau.edu.cn (F.L.); qiaojianzhong@mail.neu.edu.cn (J.Q.)

² School of Computer Science, Shenyang Aerospace University, Shenyang 110136, China; sxb@sau.edu.cn

³ Department of Computer Science, Middlesex University, London NW4 4BT, UK; x.cheng@mdx.ac.uk

⁴ College of Information, Shenyang Institute of Engineering, Shenyang 110136, China

* Correspondence: lzhaos@sau.edu.cn (L.Z.); daiqin@sie.edu.cn (Q.D.)

Received: 25 March 2020; Accepted: 15 April 2020; Published: 18 April 2020



Abstract: Effective extraction of human body parts and operated objects participating in action is the key issue of fine-grained action recognition. However, most of the existing methods require intensive manual annotation to train the detectors of these interaction components. In this paper, we represent videos by mid-level patches to avoid the manual annotation, where each patch corresponds to an action-related interaction component. In order to capture mid-level patches more exactly and rapidly, candidate motion regions are extracted by motion saliency. Firstly, the motion regions containing interaction components are segmented by a threshold adaptively calculated according to the saliency histogram of the motion saliency map. Secondly, we introduce a mid-level patch mining algorithm for interaction component detection, with object proposal generation and mid-level patch detection. The object proposal generation algorithm is used to obtain multi-granularity object proposals inspired by the idea of the Huffman algorithm. Based on these object proposals, the mid-level patch detectors are trained by K-means clustering and SVM. Finally, we build a fine-grained action recognition model using a graph structure to describe relationships between the mid-level patches. To recognize actions, the proposed model calculates the appearance and motion features of mid-level patches and the binary motion cooperation relationships between adjacent patches in the graph. Extensive experiments on the MPII cooking database demonstrate that the proposed method gains better results on fine-grained action recognition.

Keywords: fine-grained action recognition; motion saliency; mid-level patch; object proposal

1. Introduction

In recent years, fine-grained action recognition has attracted a substantial amount of research interest, as it plays an important role in human–computer interaction, smart homes, elderly/child care, medical surveillance, and robots [1–8]. However, most existing action recognition methods focus on coarse-grained actions, such as full-body activities in daily life, e.g., jumping and waving. Compared with coarse-grained action recognition, fine-grained action recognition is more challenging due to the complex human motions and interactions, small body movements, large intra-class variability, small inter-class variability, etc. [1,9]. Therefore, fine-grained action recognition is still in its infancy.

In the motion process of fine-grained actions, human body parts always interact with operated objects that are almost small and have a large variety of categories [10,11]. Furthermore, there are many occlusions on body parts and operated objects. In the process of interactions, the same action

operated by different people may show different looks and motion distributions because everyone has a different operation method. Meanwhile, most motions of fine-grained actions are centered at terminal joints, such as hands. Take the large fine-grained action database MPII cooking database as an example [1]: the action “cut” can be subdivided into “cut apart”, “cut dice”, and “cut slices”. Figure 1a–c are “cut apart”, “cut dice”, and “cut slices” actions that are operated by the same person. From Figure 1, it can be seen that the appearance and operated objects of these three actions are very similar. Figure 1c,d illustrate the “cut slice” action operated by different people, and it can be seen that the same action performed by different people does not look the same. Figure 1 also shows that each of these fine-grained actions is often characterized by a complex distribution of operated objects (e.g., a knife and a cucumber) and body parts (e.g., arms and hands) along with their interactions. Therefore, how to partition body parts and operated objects and model interactions between them (spatial context information) play important roles in fine-grained action representation and recognition.

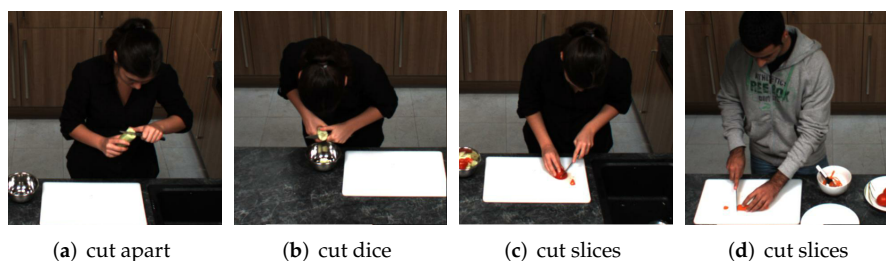


Figure 1. Some exemplars of fine-grained actions in MPII cooking database. (a) cut apart, (b) cut dice, (c) cut slices, and (d) cut slices.

There are many algorithms of fine-grained action recognition [12–18]. However, recent approaches consider the video as a whole [1,12,19]. The disadvantage of these algorithms is that they may lead to the loss of image spatial information. In order to model the human–object interaction, some approaches [9,17,18,20] detect objects involved in actions. Because fine-grained actions often involve large kinds of operated objects and the training of object detectors requires extensive manual annotation, the scalability of these algorithms is poor. In addition, the performance of the algorithms depends on the accuracy of object detection, which makes them unstable. Other algorithms employ mid-level patches to represent actions [10,11,15,21]. This kind of algorithm decomposes actions into mid-level patches with different levels and granularities and uses these patches to describe the complex structure of actions in space. The problem is that they generate a large number of object proposals by fixed segmentation grids of the whole image [16,22], which can reduce the discriminative ability of features [23,24]. Furthermore, these algorithms would extract all static objects and dynamic objects in the scenario, while the static ones are unnecessary for action recognition.

To solve these problems, this paper presents a fine-grained action recognition method based on motion saliency and mid-level patches. The framework of the algorithm is shown in Figure 2. Firstly, the frame is converted into a motion saliency map by computing motion saliency for videos [25]. The interaction components are salient in the motion saliency map because their motions are different from their neighbors. Therefore, we can extract the motion regions automatically by a threshold-based method. Secondly, the partitioned motion saliency regions are single grained, and they contain limited effective information. Therefore, they are not suitable for mid-level patch (which represents the interaction part) mining. In order to extract multi-granularity object proposals, we merge the motion regions by the idea of the Huffman algorithm. The mid-level patch detectors are then trained with K-means clustering and SVM [15] for these proposals in an unsupervised way. Finally, our fine-grained action recognition model utilizes a graph structure to describe the interaction relationships between mid-level patches. The DT (Dense Trajectory) feature [12,19] is adopted to extract region features. Besides the appearance and motion features, the binary motion cooperation relationship between different mid-level patches is calculated to obtain spatial context information, which includes motion

consistency and trajectory consistency [21]. SVM is used for action recognition with these features. The main contributions of our algorithm are as follows.

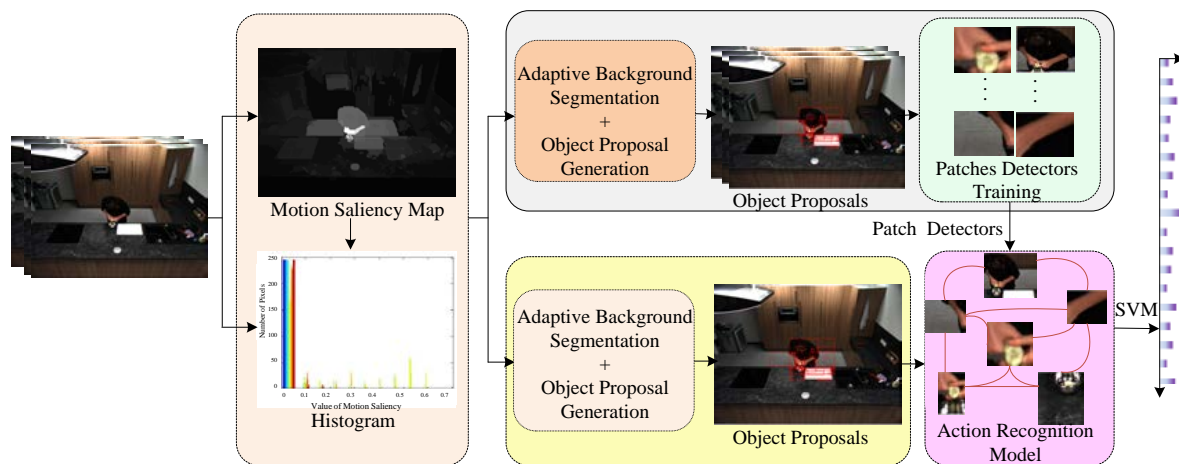


Figure 2. Our fine-grained action recognition method, which includes motion region segmentation based on motion saliency, object proposal generation by the idea of the Huffman algorithm, mid-level patch detector training with K-means clustering and SVM, and action recognition with a graph structure.

(1) We propose a threshold-based motion region segmentation algorithm for motion saliency maps without manual annotation, which can naturally consider the static regions and objects as background. The threshold is adaptively calculated according to the histogram of the saliency map. These motion regions will act as object proposals for mid-level patch mining, which can significantly reduce the number of object proposals and provide more reliable ones.

(2) In order to extract multi-granularity object proposals, an object proposal generation algorithm by the idea of the Huffman algorithm is proposed to merge the original motion saliency regions with the greedy strategy. All nodes in the Huffman tree are regarded as object proposals, which are capable of capturing different aspects of actions ranging from the fine-grained body parts to the large chunks of human–object interactions.

(3) A fine-grained action recognition model is proposed and the interaction relationship between mid-level patches in the action is described with a graph structure.

The rest of this paper is organized as follows. In Section 2, we review the existing related work. The adaptive segmentation of the motion regions and the mid-level patch mining algorithm are described in detail in Section 3. The action recognition model based on mid-level patches is described in Section 4. In Section 5, experimental results are given and analyzed. The conclusion is given in Section 6.

2. Related Work

The studies on action recognition are numerous, such as fine-grained action recognition [1,2,9,10,26,27] and coarse-grained action recognition [28,29]. In the following, we review the work that is most related to the current study.

Rohrbach et al. [1] presented a database on kitchen operations to evaluate fine-grained action classification. It is composed of 65 different actions that take place continuously within 8 hours of recording. This database is mainly used for fine-grained action recognition [3,9–11,26,30–33]. In the literature [1], two schemes for action recognition are proposed, one is based on the human pose and the other is based on the DT feature. On the basis of these and upon further study, Rohrbach et al. [27] proposed a hand-centric approach for fine-grained activity classification and composite activity recognition. Yang et al. [9] proposed a representation and classification pipeline that seamlessly incorporated localized semantic information for fine-grained action recognition. However,

these methods [1,9,27] require explicit object detection or pose estimation. Interaction Part Mining [10] involves a fine-grained action recognition pipeline by mid-level part mining. It applies Max-N pooling to keep important candidate parts, which would generate redundant information.

Following the impressive results of deep learning on the task of image classification [34] and other research fields [35,36], many efforts have been made to train deep networks for the task of action recognition. Fernando et al. [3] proposed a temporal pooling function, which uses the evolution of video temporal structure to represent the videos. Li et al. [30] proposed a new network structure, which encodes spatiotemporal features by sharing weights between the learned parameters. Ni et al. [2] proposed an end-to-end fine-grained action detection system based on RNNs (Recurrent Neural Networks), which parses the interactive objects frame by frame in a video to describe a specific action for fine-grained action recognition. These algorithms achieved good recognition effects on the fine-grained actions while fusing the DT features and the CNN (Convolutional Neural Network) features, but the results of using the two features respectively were not satisfactory.

There is work that utilized spatiotemporal segments to capture different interaction components of human action and represent videos [11,37,38]. Some work [9,10] focused on the problem of modeling context information between actions and objects and investigated how to parse or track operated objects. Weinzaepfel et al. [39] detected proposals at the frame-level and scored them with static and motion CNN features, and then tracked high-scoring proposals throughout the video to generate spatiotemporal proposals. Multi-level representation of action is popular in action recognition because it can extract the participation components of action in different granularities [40]. Lan et al. [11] extracted mid-level elements from region proposals to represent actions. Ma et al. [41] proposed a hierarchical spatiotemporal segmentation method for action recognition and location. Spatiotemporal segments included the whole human body and body parts with different granularities. The action participation segments-based method is also used in this paper. Our mid-level patches include human body parts and motion objects that participate in action, and the static objects can be excluded based on motion saliency.

Mid-level patch mining employs object proposals to guide the detection of patches. Selective Search [22] first partitions the image into regions and greedily merges these regions to generate proposals according to feature similarities between regions. Bing [16] uses a simple linear classifier to calculate scores for a large number of candidate windows based on the proposed feature "BING". EdgeBoxes [42] defines an edge-based function to grade the score for any candidate bounding box according to edge response. Furthermore, MCG (Multiscale Combinatorial Grouping) [43], Multibox [44], and other algorithms [45] provide alternative proposal methods. In contrast, our object proposals are generated based on the motion saliency and the adaptive background segmentation.

3. Mid-Level Patch Mining Based on Motion Saliency

To benefit from the space context, spatial pooling divides a video using fixed segmentation grids and pools the features locally in each grid cell [22–24]. Though the performance has been improved, different action instances of the same category with various human localizations in spatiotemporal volume [46,47] can result in a non-uniform distribution of features. Furthermore, one interaction component may be divided into different cells due to the fixed segmentation. Therefore, the key problem is how to divide the interaction components accurately in different instances. To avoid manual annotation, we use mid-level patches (detected from object proposals) to represent the interaction components. Most algorithms apply Bing [16] or Selective Search [22] to extract object proposals in a fixed way. All the objects in the scene are regarded as proposals, including dynamic ones and static ones unrelated to the action. In our algorithm, object proposals are extracted based on the motion saliency [25], which can automatically exclude the static regions.

3.1. Motion Saliency-Based Motion Region Partition

We define the motion saliency as a mapping $s: P \rightarrow \mathbb{R}$, where P is a spatial partition of an image. $p \in P$ is a region of the image and $s(p)$ gives the motion saliency value of the region. P can be obtained from any spatial partition of the image. The value of $s(p)$ is in the range $[0,1]$. We adopt the spatiotemporal saliency computation method [25] to calculate the motion saliency. First, the image is divided into several spatiotemporal regions P , and the motion saliency value $s(p)$ is calculated for each region p . Therefore, the image is converted into a motion saliency map according to the motion saliency value of each region [25], shown in Figure 3a,b presents the maps after motion region segmentation. In Figure 3a,b, the salience value $s(p)$ of each region p is multiplied by a coefficient within 5 in order to visualize the motion saliency maps with better contrast. In motion saliency maps, the interaction components with high motion saliency value are all highlighted, and the differences between the motion regions and the static regions are obvious. Figure 3c is the visualization of motion saliency regions in the original RGB images after motion region segmentation. Different from motion saliency maps, RGB images show the region boundary.

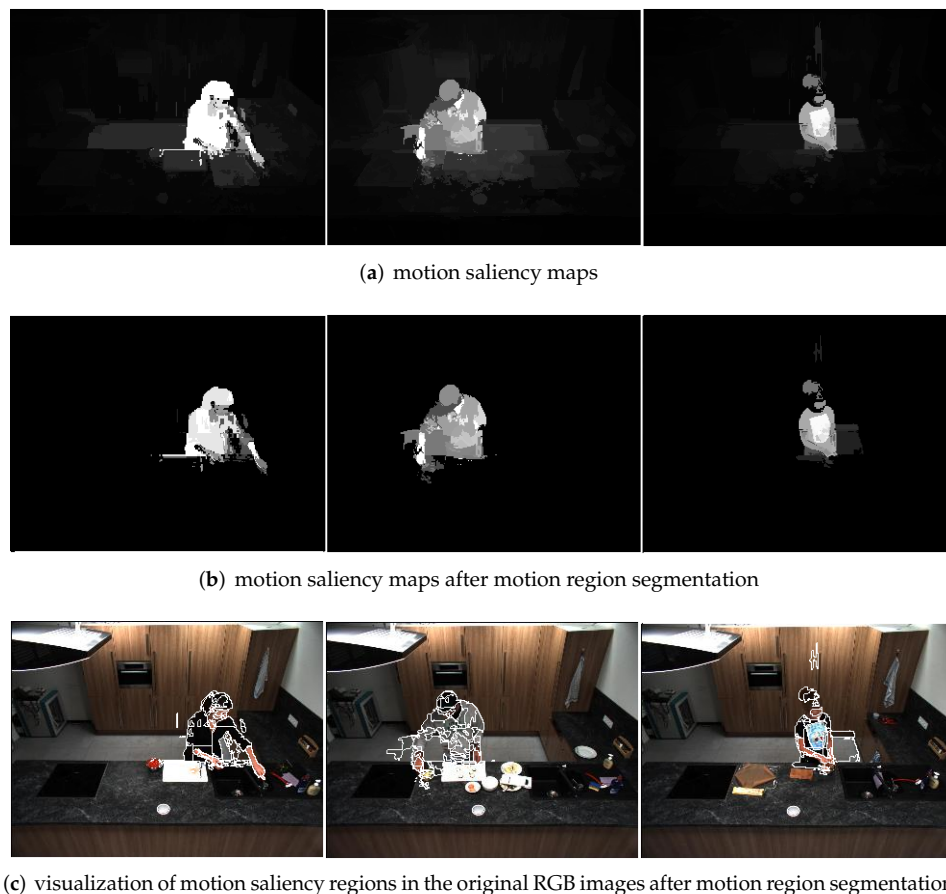


Figure 3. Results of the motion saliency region partition. The action categories of these three groups of images are “wash hands”, “put in bowl”, and “peel”.

3.2. Adaptive Motion Region Segmentation

The motion saliency computation is of the whole frame, and the real motion saliency regions still need to be segmented. We only need the motion regions. In other words, we want to segment the regions including human body parts and objects involved in the action. Traditional background segmentation methods cannot meet this requirement. In this paper, motion region segmentation is based on the motion saliency.

As shown in Figure 3a, the motion saliency map is a gray image. For this kind of image, the threshold-based method is the most commonly used background segmentation algorithm. The motion saliency value distributions of different images are inconsistent so that a fixed threshold is not suitable for all images. Figure 4 shows the visualization of histograms of the motion saliency maps. As shown in Figure 4, the histogram of the motion saliency map is discrete, and the first bin in the histogram contains most of the static regions. The threshold of motion region segmentation can be adaptively calculated, which takes the median of the first two bins in the histogram. That is to say, $Threshold = (Saliencyhist(1) + Saliencyhist(2)) / 2$. $Saliencyhist$ represents the central numerical vector of all bins in the histogram, and the threshold is calculated for each image separately. The segmentation results are displayed in Figure 3b, which shows that any region in which motion saliency value $s(p)$ is larger than the $Threshold$ would be reserved as foreground.

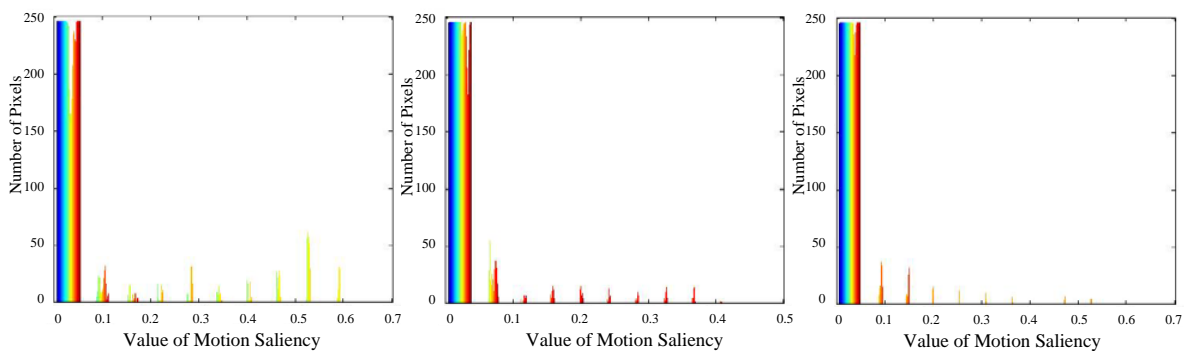


Figure 4. Histograms of motion saliency maps.

3.3. Object Proposal Generation by the Idea of the Huffman Algorithm

In Figure 3c, the human body is divided into several small motion saliency regions. The granularity of these regions is single, and these regions contain limited information and are not suitable as the elements for extracting mid-level patches. Therefore, we use a region merging algorithm to obtain multi-granularity motion regions [48,49]. The Huffman algorithm is a simple and efficient lightweight algorithm. Inspired by the Huffman algorithm, we propose a region merging algorithm to merge the motion saliency regions with the image appearance features and the motion saliency features. All node regions in the Huffman tree are regarded as object proposals, which can obtain multi-granularity region partition results [50]. The initial region partition result of the foreground is $P^0 = \{p_1^0, p_2^0, \dots, p_n^0\}$. For each region merging iteration i , two adjacent regions with the highest similarity are merged to obtain a new non-terminal node. The two merged regions are deleted from P^i and the new node is added to P^i . The algorithm continues to iterate until the Huffman tree is built or there are no adjacent regions that have not been merged.

In order to describe the characteristics of different aspects of these regions more detailed, we choose a variety of features, including color similarity, motion saliency similarity, and region size factor. The feature values are in the range $[0, 1]$ to facilitate merging.

(1) Color Similarity

In order to account for the different scenes and lighting conditions, we describe the motion saliency regions in a variety of color spaces, including RGB, Lab, HSV, normalized RGB (nRGB), and rgi [22]. Experimental results show that the nRGB color space has the best discrimination, where it is chosen as the color space. The color feature of a motion region is represented by a color histogram, and each color channel in the histogram has k bins. In order to extract image features from different scales, k can take multiple values similar to image pyramids. The color histogram of the i th region p_i is $C_i = \{c_i^1, \dots, c_i^m\}$, and $m = k * 3$ for color images. The color histograms are normalized with the $L1$ norm. The color similarity between region p_i and p_j is measured by the histogram intersection as

$$P_{color}(i, j) = \sum_{l=1}^m \min(c_i^l, c_j^l). \quad (1)$$

(2) Motion Saliency Similarity

The motion saliency similarity between two adjacent motion regions is also measured by histogram intersection. The motion saliency map only has one channel, and the motion saliency histogram of the i th region p_i is $MS_i = \{ms_i^1, \dots, ms_i^m\}$. The motion saliency similarity between region p_i and p_j is shown as

$$P_{motion}(i, j) = \sum_{l=1}^m \min(ms_i^l, ms_j^l). \quad (2)$$

(3) Region Size Factor

The reason for introducing region size factor is to ensure that the small regions could be merged early. The calculation method is as

$$P_{size}(i, j) = 1 - \frac{size(p_i) + size(p_j)}{size(im)}, \quad (3)$$

where $size(im)$ is the size of the image im , and $size(p_i)$ is the size of the region p_i .

Our final similarity $P_{i,j}$ between region p_i and p_j is the combination of these three factors as

$$P_{i,j} = \alpha_1 P_{colour}(i, j) + \alpha_2 P_{motion}(i, j) + \alpha_3 P_{size}(i, j), \quad (4)$$

where $\alpha_i \in [0, 1]$, $\sum \alpha_i = 1$, and α_i denotes the weight of the factor i . In the experiment, all factors have equal weights.

The number of motion saliency regions in each frame is uncertain. If there are n regions, then the similarity measure calculation between neighbor regions could be $\frac{n(n-1)}{2}$ times. However, only adjacent regions can be merged, which means that only the similarity between adjacent regions needs to be calculated. If the average number of neighborhoods of each region is n' , the number of pairs to be calculated is $n \times n'$. In most cases, $n' \ll n$ can greatly mitigate the computation costs. Furthermore, a new non-terminal node in each iteration only affects the surrounding regions so that only information about these adjacent nodes needs to be updated. The variables involved in the algorithm are as follows. i and j are the numbers of two adjacent regions, respectively.

A_i denotes the set of object proposals adjacent to region p_i , and D_i represents the properties of p_i , i.e., the region size, the color, and the motion saliency. $P_{i,j}$ denotes the similarity between the object proposal p_i and p_j , where $p_j \in A_i$, and vice versa.

For each step of the region merging, two regions with the highest similarity in all adjacent region pairs are selected where the similarity is related to D . The object proposal generation algorithm is shown in Algorithm 1.

Figure 5 shows the object proposal extraction results of our algorithm and the Selective Search algorithm [22]. It can be seen from the figure that the two algorithms can effectively extract the interaction regions. Both of them first partition the image into candidate regions, and then merge these regions with a Huffman-like algorithm. However, our algorithm generates the candidate regions based on the motion saliency that has a natural advantage in action recognition. Most object proposals segmented by the two algorithms are the same, but the regions in this paper are more accurate. For example, the motion information of the regions are stronger and the static objects are basically excluded. Take the action “wash hands” in Figure 5 as an example, the object “knife” is irrelevant to the action. In the first image in Figure 5b, the object “knife” is regarded as background by motion region segmentation with the proposed method. In the first image in Figure 5c, the object “knife” is extracted as an object proposal with the Selective Search [22]. Furthermore, this object will still be

regarded as foreground even with traditional background segmentation methods [51] because the object “knife” does not belong to the background.

Algorithm 1 Object proposal generation algorithm.

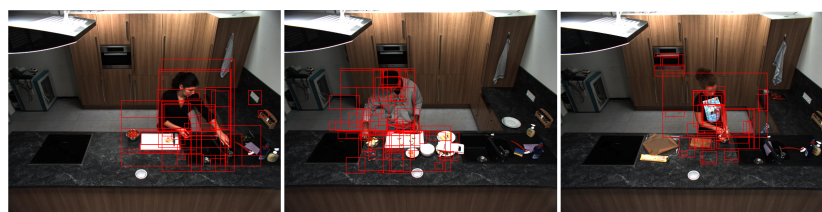
- 1: Initialization: The segmentation threshold is calculated according to the motion saliency map, and initial motion saliency regions in the foreground image are $P^0 = \{p_1^0, p_2^0, \dots, p_n^0\}$;
 - 2: $\forall p_i^0 \in P^0$, compute D_i and A_i ;
 - 3: Compute the similarity between adjacent regions, $PS = \{P_{i,j} | \forall p_i \in P^0 \text{ and } p_j \in A_i \text{ and } i > j\}$;
 - 4: $k = 0$; $m = n$; $regionset = P^0$; // m is the number of the current object proposals;
 - 5: **while** ($len(P^k) > 1$) & (*if exist* $A_i \neq \emptyset, p_i^k \in P^k$) **do**
 - 6: $k = k + 1$;
 - 7: $m = m + 1$;
 - 8: $P_{i,j} = \max(PS)$;
 - 9: $P^k = (P^{k-1} \cup \{p_m\}) - \{p_i^{k-1}, p_j^{k-1}\}$;
 - 10: compute D_m according to D_i and D_j ;
 - 11: $A_m = A_i \cup A_j - \{p_i^{k-1}, p_j^{k-1}\}$;
 - 12: $\forall p_v^{k-1} \in A_m, A_v = (A_v \cup \{p_m\}) - \{p_i^{k-1}, p_j^{k-1}\}$;
 - 13: $PS = (PS \cup \{P_{m,o} | p_o \in A_m\}) - \{P_{u,v} | u, v = i \text{ or } j\}$;
 - 14: **end while**
-



(a) original images



(b) object proposals extracted by the proposed algorithm



(c) object proposals extracted by the selective search algorithm

Figure 5. Results of object proposal extraction of our algorithm and Selective Search. The red boxes represent the object proposals. The action categories of these three groups of images are “wash hands”, “put in bowl”, and “peel”.

3.4. Unsupervised Mid-Level Patch Detector Training

The composition of fine-grained action is complex. Take the action “taking things from the fridge” as an example. The human body parts involved in the action are the arms and hands, and objects involved in the action are a fridge and other objects. In order to describe the action more reasonably and in accordance with Huang et al. [52], we divide the action into different interaction components.

In space, action can be expressed as the interaction of different body parts and operated objects. Many algorithms extract these interaction components with annotations and pre-training, which require large manual annotation. Furthermore, most algorithms only consider human body parts, while the objects involved in the action are usually excluded because of the diversity of categories and the complexity of annotations. This paper adopts mid-level patches to represent the interaction components, including body parts and operated objects, and using an unsupervised method with K-means clustering and SVM to train patch detectors for object proposals [15].

The proposed algorithm trains multiple patch detectors for each action category. Firstly, K-means or other clustering methods are used to cluster object proposals obtained as above. In order to ensure the purity of each cluster, the number of clusters should be large enough. However, running K-means on the object proposals does not produce good clusters. Therefore, SVM is combined with the clustering process, and the classifier is trained for each cluster generated by clustering. In order to ensure the effectiveness of mid-level patch detectors, the clustering and training of classifiers are iterated repeatedly until convergence. The object proposals in images of the same action category in the training set are taken as positive samples, while the object proposals in other categories are taken as negative samples.

In order to ensure that the detected mid-level patches can cover all the interaction components as far as possible, multiple patch detectors are trained for each action category. Finally, 15 detectors are selected, and some detectors are selected manually. The mid-level patches detected by the detectors form the mid-level representation of the action.

4. Action Recognition with Graph Structure

Most human actions can be regarded as the process of interaction between human body parts and operated objects (if any) in space. Reasonable descriptions and modeling of the above interaction components and their spatial relationships can recognize the action more effectively. In this paper, the interaction components are represented by mid-level patches detected by the patch detectors, and their space relationships are modeled by a graph structure.

4.1. The Graph Structure

In order to effectively capture the spatial context information during the action process, a graph structure is used to represent the relationship between mid-level patches [53,54]. Firstly, all mid-level patches are constructed into an undirected action graph $G = (V, E)$, where the node $i \in V$ represents a mid-level patch, and the edge $(i, j) \in E$ represents the relationship between two patches. Initially, all nodes form a fully connected graph.

In fact, not all the body parts have a cooperative relationship during the action process. In most cases, cooperation only involves parts of human body parts. Using a fully connected graph to represent the relationship between the patches will generate many redundant relationships and increase computation costs. Therefore, this paper adopts a graph segmentation algorithm to segment the action graph G and deletes unnecessary edges in which two endpoints have no cooperation relationship [55,56]. For any two nodes i and j , if two mid-level patches represented by the nodes move simultaneously for long enough, they are considered to have a motion cooperation relationship. Otherwise, there is no cooperation relationship between them.

Suppose that n mid-level patches are captured in frame k , and they are expressed as $F_k = (f_k^1, f_k^2, \dots, f_k^n)$. For each patch f_k^i , we extract the DT feature. The motion trajectories of all

feature points in the patch f_k^i (absolute coordinate in the image) are expressed as $tc_i = \{tc_i^m\}$. $tc_i^m = \{tc_i^m [t]\}_{t=ts}^{td}$, where the vector $tc_i^m [t]$ represents the coordinate vector (x, y) of feature point m at time t , and $[ts, td]$ is the time range of the trajectory of that point.

If two patches are both in motion for at least δ frames, it is considered that there exists a motion cooperation relationship between them. We set $T_i = \bigcup_{\forall m \in f_k^i} [ts_m, td_m]$, and the adjacency matrix M_k of graph G in frame k is defined as Equation (5). T_i is the time interval in which patch i is in motion.

$$M_k(i, j) = \begin{cases} 1, & \text{if } |T_i \cap T_j| > \delta \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

4.2. Motion Cooperation Relationship of Mid-Level Patches

The key problem is how the relationship between mid-level patches can be calculated. We adopt the appearance and motion features to describe the patches. Furthermore, the relationship between any two related patches is described by the binary motion cooperation relationship.

The feature term of a patch is $u_i = \omega_i \cdot x_i$, which uses SVM to calculate the score of patch i . The DT feature x_i is used to describe the patch i , and the parameter ω_i is obtained by learning.

The binary relation term $u_{i,j}$ describes the spatial motion cooperation relationship between different patches i and j . This paper calculates the relationship between mid-level patches from two aspects, which are trajectory consistency and motion consistency.

(1) The Trajectory Consistency

For each patch i , the mean patch trajectory is used to describe its motion trajectory, and it is defined as

$$pt_i [t] = \frac{1}{|\{tc_i\}|} \sum_{\forall m, t \in [ts_m, td_m]} tc_i^m [t], t \in T_i. \quad (6)$$

In order to describe the change of trajectories of two patches effectively, the $pt_i [t]$ and $pt_j [t]$ are further processed [21]. First, the canonical relative position is estimated, $d_{f_k^i, f_k^j} [t] = |pt_i [t] - pt_j [t]| / \sigma$, where σ is a scale variable. $d_{f_k^i, f_k^j} [t]$ takes a value only in the coexisting time interval. Subsequently, a feature capturing the rate of the convergence and divergence of the trajectory pairs is calculated, where $v^{x,y} [t] = d_{f_k^i, f_k^j} [t + 1] - d_{f_k^i, f_k^j} [t]$ [21]. The relation between the two patches is symmetric. For convenience, we quantize two elements x and y in the vector $v^{x,y} [t]$ of any two related patches i and j at time t , respectively. The quantized vector is as follows:

$$\begin{aligned} \varphi^{i,j} (x) &= [v_{-1} (x), \rho_{-\eta} (x), \dots, \rho_0 (x), \dots, \rho_\eta (x), v_1 (x)], \\ v_{\pm 1} (x) &= \left(1 + e^{\frac{\pm x + \mu_0}{s_0}} \right)^{-1}, \rho_d (x) = e^{-\frac{(x - \mu_d)^2}{s_d^2}}, d \in [-\eta, \dots, 0, \dots, \eta]. \end{aligned} \quad (7)$$

Equation (7) is performed on the average trajectories pt_i and pt_j in the time interval $T_i \cap T_j$, and the feature vector $\phi_{i,j} = (\varphi^{i,j} (x), \varphi^{i,j} (y))$ is obtained, where $\phi_{i,j} \in \mathbb{R}^{((2\eta+3) \times \{T_i \cap T_j\}, 2)}$. The parameters in Equation (7) are set as Raptis et al. [21], where $\eta = 3$. s_0 and μ_0 are set so that the sigmoidal functions v can cover the extremes of the domain. In Gaussian functions ρ , μ_d is set to cover the relative velocity domain spanned by the database, and s_d changes along with μ_d according to the variance formula. The feature vector $\phi_{i,j}$ can better describe the trajectory consistency between the two mid-level patches i and j [21].

(2) Motion Consistency

Motion consistency is calculated by feature histograms of HOG (Histograms of Oriented Gradients), HOF (Histograms of Optical Flow), and MBH (Motion Boundary Histograms, MBHx and MBHy). For the i th patch f_k^i in the frame F_k , the feature histogram is $H_k^i = \{h_i^c\}_{c=1}^4$, where c represents the four different features and h_i^c is the Bow of the c th feature of patch i . The motion consistency s_{ij} between any two patches i and j is as follows:

$$s_{ij} = \sum_c \omega^c \cdot \exp \left(-\frac{dc(h_i^c, h_j^c)}{B^c} \right), \tag{8}$$

where B^c is used to normalize different channels, and its value is the average of Euclidean distance between the histograms of characteristic c in the training set; $dc(h_i^c, h_j^c)$ denotes the Euclidean distance between two groups of feature vectors; and ω_c is obtained by training.

According to the analysis above, the binary motion cooperation relationship between any two patches i and j is as follows:

$$u_{i,j} = \omega_{i,j}^{p_1} \cdot \phi_{i,j} + \omega_{i,j}^{p_2} \cdot s_{i,j}. \tag{9}$$

4.3. Action Recognition Model

Our goal is to accomplish action recognition based on feature vector and action label. The action score of each video clip v is determined by Equations (10) and (11). x_0 represents the global feature.

$$S_v = \sum_i \omega_i \cdot x_i + \sum_i \sum_j \left(\omega_{i,j}^{p_1} \cdot \phi_{i,j} + \omega_{i,j}^{p_2} \cdot s_{i,j} \right) + \omega_0 \cdot x_0. \tag{10}$$

In this paper, weak supervision is used to train the fine-grained action recognition model. For any video v , only the label Y of the action is known. The mid-level patches are detected by patch detectors. SVM with a chi-square kernel is used for model training. The action recognition model is as follows:

$$\min_{\omega, \xi \geq 0} \frac{1}{2} \|\omega\|^2 + C \sum_n \xi_n, \tag{11}$$

$$S_{v_n}(X, Y) - S_{v_n}(X, Y^*) \geq \Delta_{0/1}(Y, Y^*) - \xi_n, \forall n.$$

In the model, a 0-1 loss function is used to estimate the difference between the ground-truth label Y and the label Y^* predicted by the action recognition model.

5. Experiments

This paper validates the proposed action recognition model on the MPII cooking database [1], including 65 fine-grained kitchen actions. It contains 44 videos in total completed by 12 different people (or equally 5609 video clips, and 881,755 frames). The main difficulty of this database is that the difference among categories is small, but the difference between the same category instances is quite large. Moreover, the background of the kitchen in the video is dark, which affects the detection accuracies of the human body and objects.

Experiment settings: Following Rohrbach et al. [1], we perform leave-one-person-out cross-validation for performance evaluation. We adopt 5 subjects to train the model and the remaining 7 subjects to test in 7 cross-validation rounds. We use one-vs-all SVM with a chi-square kernel. For codebook training, the number of cluster centers is 4000. m in Equation (1) is 75; in Equation (2), it is 25. We evaluate classification performance in terms of multi-class precision (Pr), recall (Rc), and mean average precision (mAP).

5.1. Parameter Selection for Object Proposal Generation

In order to adapt to different scenes and lighting conditions, we perform object proposal generation on a variety of color spaces. The color spaces are chosen to be the same as with Selective Search [22]. We test the performance of these color spaces by classifying actions with the last four regions in the Huffman tree. In this group of experiments, the HOG in the DT feature is adopted. Table 1 shows that the nRGB color space has the best mAP of 54.4%. The second one is RGB. According to the experiment results, we choose the nRGB to calculate the color similarity.

Table 1. Recognition accuracies for different color spaces (%).

Color Space	mAP(%)
RGB	53.7
nRGB	54.4
Lab	53.5
rgI	52.1
HSV	47.6

Table 2 shows the recognition results with or without the region size factor in Equation (4). The second to fourth columns in Table 2 are weights of the color similarity, the motion saliency similarity, and the region size factor, respectively. All factors have equal weights. Based on Table 2, the result of region merging is better with the region size factor.

Table 2. Recognition accuracies with or without region size factor (%).

With or Without Size Factor	Weight of Color Space	Weight of Saliency	Weight of Size	mAP(%)
with size factor	0.33	0.33	0.33	54.4
without size factor	0.5	0.5	0	49.1

5.2. Action Recognition Accuracies of Different Features

We present the action recognition results of different features on the MPII cooking database in Table 3. The features are HOG, HOF, MBHx, MBHy, MBH, and the combination of them. Combining all features achieves the best result and significantly improves the mAP by more than 3% compared with other features.

Table 3. Recognition accuracies of different features (%).

Feature	Pr	Rc	mAP
HOG	43.5	38.2	59
HOF	45.5	40.8	59.4
MBHx	45.2	39.7	61.2
MBHy	45.3	41.5	66.6
MBH	49.0	42.5	67.4
Combined	52.0	46.0	69.8

5.3. Comparison with State-of-the-Art Methods

Table 4 presents the experimental results of our fine-grained action recognition algorithm against the state-of-the-art methods. From Table 4, we can see that the proposed method performs well. On the MPII cooking database, our method shows an overall mAP of 69.8%, which is significantly better than the baseline [1] (57.9). The proposed method outperforms the Hierarchical Mid-Level Actions [11], Interaction Part Mining (Max Pooling) [10], IDT-FV (Improved Dense Trajectory, Fisher Vector) [31], P-CNN (Pose-Based CNN) [31], Binary Hashing CNN Features [32], and the Order-Constrained Kernelized Feature [33]. Our algorithm is comparable to the Semantic Features [9] and Interaction

Part Mining (Max-N Pooling) [10] methods. The Semantic Features method [9] requires object detection, while our method partitions object proposal regions in motion saliency maps without manual annotation. Interaction Part Mining (Max-N Pooling) [10] detects N instances for each patch detector to gain better results with abundant information, whereas our method only detects one instance for each detector. The per-class classification results for these three algorithms on the MPII cooking database are shown in Figure 6 (only 54 categories are shown).

Table 4. Comparison of action recognition accuracies of different methods on the MPII cooking database.

Method	mAP(%)
Holistic + Pose [1]	57.9
Holistic Dense Trajectories [1]	59.2
Hierarchical Mid-Level Actions [11]	66.8
Interaction Part Mining (Max Pooling) [10]	69.1
Interaction Part Mining (Max-N Pooling) [10]	72.4
Semantic Features [9]	70.5
IDT-FV [31]	67.6
P-CNN [31]	62.3
Binary Hashing Convolutional Neural Network (CNN) Features [32]	63.8
Order-constrained Kernelized Feature [33]	53.0
Our Model	69.8

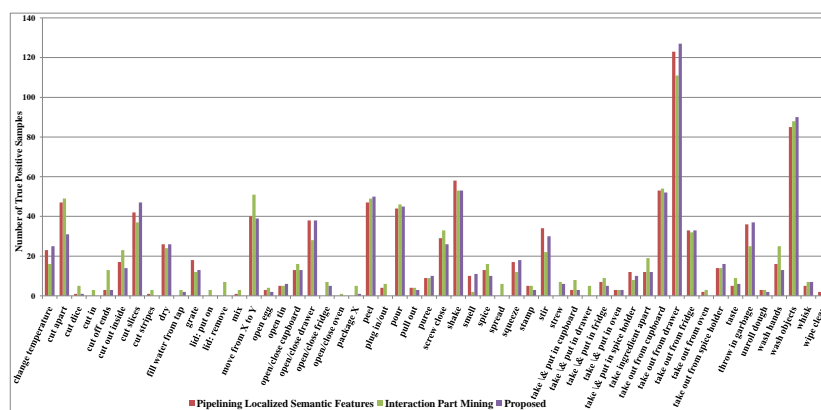


Figure 6. Per-class classification performance comparison on the MPII cooking database. We compare our method with the Semantic Features and Interaction Part Mining (Max-N Pooling) methods in terms of the number of true positive predictions.

We compare our method with the Semantic Features [9] and Interaction Part Mining (Max-N Pooling) [10] methods. As shown in Figure 6, the proposed algorithm achieves better or comparable results for most categories of actions. For actions with visible operated objects, such as “change temperature”, “cut slices”, and “open tin”, the recognition accuracies of our algorithm are the best. Meanwhile, our method and the Semantic Features method are comparable for other categories. Unlike the Semantic Features method, our algorithm does not require manual annotation, which shows that the segmentation of interaction components based on motion saliency is effective. For actions in which operated objects are difficult to detect, such as “cut dice”, and “cut in”, the results of the three algorithms are comparable. When the category of action has more samples, our method and the Semantic Features method perform well, such as “take out from drawer” and “wash objects”. When the category has fewer samples, the recognition effect of Interaction Part Mining (Max-N Pooling) is good, such as “cut dice” and “lid: remove”, but the advantage is not obvious. The reason may be that Max-N Pooling can extract more effective information for small sample categories, but our algorithm can obtain more effective information when the number of samples is enough. That is to say, our algorithm

may have a better result than Interaction Part Mining (Max-N Pooling) with enough samples. Figure 6 verifies the proposed method based on motion saliency and mid-level patches.

6. Conclusions

In this paper, a fine-grained action recognition algorithm based on motion saliency and mid-level patches is proposed. Motion regions are adaptively extracted in the motion saliency maps without manual annotation. With the object proposal generation algorithm, multi-granularity object proposals based on motion saliency regions are obtained, and the mid-level patch detectors are trained in an unsupervised way. Action recognition is implemented by calculating the appearance and motion features of patches and the binary motion relationship between patches in the action graph. The experiment results show that the proposed fine-grained action recognition algorithm works effectively. The extracted mid-level patches can cover most interaction components (including human body parts and objects), and their representation ability is strong. The motion cooperation relationship-based action recognition model can classify the actions effectively. In order to further improve recognition accuracy, the human body parts and operated objects should be extracted and located more accurately.

Author Contributions: All authors equally contributed to this work. All authors have read and agreed to the published version of the manuscript.

Funding: This paper is partly supported by the National Natural Science Foundation for Young Scientists of China (61701322), the Young and Middle-aged Science and Technology Innovation Talent Support Plan of Shenyang (RC190026), and the Liaoning Provincial Department of Education Science Foundation (JYT19052).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rohrbach, M.; Amin, S.; Andriluka, M.; Schiele, B. A database for fine grained activity detection of cooking activities. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1194–1201. [\[CrossRef\]](#)
2. Ni, B.; Yang, X.; Gao, S. Progressively Parsing Interactional Objects for Fine Grained Action Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016), Las Vegas, NV, USA, 27–30 June 2016; pp. 1020–1028. [\[CrossRef\]](#)
3. Fernando, B.; Gavves, E.; Mogrovejo, J.O.; Ghodrati, A.; Tuytelaars, T. Rank Pooling for Action Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 773–787. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Perrett, T.; Damen, D. DDLSTM: Dual-Domain LSTM for Cross-Dataset Action Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2019), Long Beach, CA, USA, 16–20 June 2019; pp. 7852–7861.
5. Cherian, A.; Gould, S. Second-order Temporal Pooling for Action Recognition. *Int. J. Comput. Vis.* **2019**, *127*, 340–362. [\[CrossRef\]](#)
6. Wang, L.; Koniusz, P.; Huynh, D. Hallucinating IDT Descriptors and I3D Optical Flow Features for Action Recognition with CNNs. In Proceedings of the International Conference on Computer Vision (ICCV 2019), Seoul, South Korea, 27 October–2 November 2019; pp. 8697–8707.
7. Ahad, M.A.R.; Antar, A.D.; Shahid, O. Vision-based Action Understanding for Assistive Healthcare: A Short Review. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, 16–20 June 2019; pp. 1–11.
8. Feng, Y.; Wu, X.; Wang, H.; Liu, J. Multi-group Adaptation for Event Recognition from Videos. In Proceedings of the 22nd International Conference on Pattern Recognition (ICPR 2014), Stockholm, Sweden, 24–28 August 2014; pp. 3915–3920. [\[CrossRef\]](#)
9. Yang, Z.; Ni, B.; Yan, S.; Moulin, P.; Qi, T. Pipelining Localized Semantic Features for Fine-Grained Action Recognition. In Proceedings of the European Conference on Computer Vision (ECCV 2014), Zurich, Switzerland, 6–12 September 2014; pp. 481–496. [\[CrossRef\]](#)

10. Yang, Z.; Ni, B.; Hong, R.; Meng, W.; Qi, T. Interaction part mining: A mid-level approach for fine-grained action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, 7–12 June 2015; pp. 3323–3331. [\[CrossRef\]](#)
11. Lan, T.; Zhu, Y.; Zamir, A.R.; Savarese, S. Action Recognition by Hierarchical Mid-Level Action Elements. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV 2015), Santiago, Chile, 7–13 December 2015; pp. 4552–4560. [\[CrossRef\]](#)
12. Wang, H.; Kläser, A.; Schmid, C.; Liu, C. Action recognition by dense trajectories. In Proceedings of the 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20–25 June 2011; pp. 3169–3176. [\[CrossRef\]](#)
13. Liu, C.; Hou, J.; Wu, X.; Jia, Y. A discriminative structural model for joint segmentation and recognition of human actions. *Multimed. Tools Appl.* **2018**, *77*, 31627–31645. [\[CrossRef\]](#)
14. Liu, C.; Wu, X.; Jia, Y. A Hierarchical Video Description for Complex Activity Understanding. *Int. J. Comput. Vis.* **2016**, *118*, 240–255. [\[CrossRef\]](#)
15. Singh, S.; Gupta, A.; Efros, A.A. Unsupervised Discovery of Mid-Level Discriminative Patches. In Proceedings of the European Conference on Computer Vision (ECCV 2012), Florence, Italy, 7–13 October 2012; pp. 73–86. [\[CrossRef\]](#)
16. Cheng, M.; Zhang, Z.; Lin, W.; Torr, P.H.S. BING: Binarized Normed Gradients for Objectness Estimation at 300fps. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014), Columbus, OH, USA, 23–28 June 2014; pp. 3286–3293. [\[CrossRef\]](#)
17. Packer, B.; Saenko, K.; Koller, D. A combined pose, object, and feature model for action understanding. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1378–1385. [\[CrossRef\]](#)
18. Prest, A.; Ferrari, V.; Schmid, C. Explicit Modeling of Human-Object Interactions in Realistic Videos. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 835–848. [\[CrossRef\]](#)
19. Wang, H.; Schmid, C. Action Recognition with Improved Trajectories. In Proceedings of the IEEE International Conference on Computer Vision (ICCV 2013), Sydney, Australia, 1–8 December 2013; pp. 3551–3558. [\[CrossRef\]](#)
20. Koppula, H.S.; Gupta, R.; Saxena, A. Learning human activities and object affordances from RGB-D videos. *J. Robot. Res.* **2013**, *32*, 951–970. [\[CrossRef\]](#)
21. Raptis, M.; Kokkinos, I.; Soatto, S. Discovering discriminative action parts from mid-level video representations. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1242–1249. [\[CrossRef\]](#)
22. Uijlings, J.R.R.; van de Sande, K.E.A.; Gevers, T.; Smeulders, A.W.M. Selective Search for Object Recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [\[CrossRef\]](#)
23. Ballas, N.; Yang, Y.; Lan, Z.; Delezoide, B.; Prêteux, F.J.; Hauptmann, A.G. Space-Time Robust Representation for Action Recognition. In Proceedings of the IEEE International Conference on Computer Vision (ICCV 2013), Sydney, Australia, 1–8 December 2013; pp. 2704–2711. [\[CrossRef\]](#)
24. Sharma, G.; Jurie, F.; Schmid, C. Discriminative spatial saliency for image classification. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3506–3513. [\[CrossRef\]](#)
25. Zhou, F.; Kang, S.B.; Cohen, M.F. Time-Mapping Using Space-Time Saliency. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014), Columbus, OH, USA, 23–28 June 2014; pp. 3358–3365. [\[CrossRef\]](#)
26. Ni, B.; Paramathayalan, V.R.; Moulin, P. Multiple Granularity Analysis for Fine-Grained Action Detection. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 756–763. [\[CrossRef\]](#)
27. Rohrbach, M.; Rohrbach, A.; Regneri, M.; Amin, S.; Andriluka, M.; Pinkal, M.; Schiele, B. Recognizing Fine-Grained and Composite Activities Using Hand-Centric Features and Script Data. *Int. J. Comput. Vis.* **2016**, *119*, 346–373. [\[CrossRef\]](#)
28. Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), San Diego, CA, USA, 20–26 June 2005; pp. 886–893. [\[CrossRef\]](#)

29. Dalal, N.; Triggs, B.; Schmid, C. Human Detection Using Oriented Histograms of Flow and Appearance. In Proceedings of the 9th European Conference on Computer Vision (ECCV 2006), Graz, Austria, 7–13 May 2006; pp. 428–441. [\[CrossRef\]](#)
30. Li, C.; Zhong, Q.; Xie, D.; Pu, S. Collaborative Spatiotemporal Feature Learning for Video Action Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2019), Long Beach, CA, USA, 16–20 June 2019; pp. 7872–7881.
31. Chéron, G.; Laptev, I.; Schmid, C. P-CNN: Pose-Based CNN Features for Action Recognition. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV 2015), Santiago, Chile, 7–13 December 2015; pp. 3218–3226. [\[CrossRef\]](#)
32. Li, W.; Feng, C.; Xiao, B.; Chen, Y. Binary Hashing CNN Features for Action Recognition. *TIIS* **2018**, *12*, 4412–4428. [\[CrossRef\]](#)
33. Cherian, A.; Sra, S.; Hartley, R. Sequence Summarization Using Order-constrained Kernelized Feature Subspaces. *arXiv* **2017**, arXiv:1705.08583.
34. LeCun, Y.; Boser, B.E.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.E.; Jackel, L.D. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput.* **1989**, *1*, 541–551. [\[CrossRef\]](#)
35. Wang, J.; Li, Y.; Shan, J.; Bao, J.; Zong, C.; Zhao, L. Large-Scale Text Classification Using Scope-Based Convolutional Neural Network: A Deep Learning Approach. *IEEE Access* **2019**, *7*, 171548–171558. [\[CrossRef\]](#)
36. Srivastava, G.; Kumar, C.V.; Kavitha, V.; Parthiban, N.; Venkataramanparthiban, R. Two-Stage Data Encryption using Chaotic Neural Networks. *J. Intell. Fuzzy Syst.* **2019**. [\[CrossRef\]](#)
37. Brendel, W.; Todorovic, S. Learning spatiotemporal graphs of human activities. In Proceedings of the IEEE International Conference on Computer Vision (ICCV 2011), Barcelona, Spain, 6–13 November 2011; pp. 778–785. [\[CrossRef\]](#)
38. Ma, S.; Zhang, J.; Ikizler-Cinbis, N.; Sclaroff, S. Action Recognition and Localization by Hierarchical Space-Time Segments. In Proceedings of the IEEE International Conference on Computer Vision (ICCV 2013), Sydney, Australia, 1–8 December 2013; pp. 2744–2751. [\[CrossRef\]](#)
39. Weinzaepfel, P.; Harchaoui, Z.; Schmid, C. Learning to Track for Spatio-Temporal Action Localization. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV 2015), Santiago, Chile, 7–13 December 2015; pp. 3164–3172. [\[CrossRef\]](#)
40. Lan, T.; Chen, L.; Deng, Z.; Zhou, G.; Mori, G. Learning Action Primitives for Multi-level Video Event Understanding. In Proceedings of the Computer Vision—ECCV 2014 Workshops, Zurich, Switzerland, 6–7 and 12 September 2014; pp. 95–110. [\[CrossRef\]](#)
41. Ma, S.; Zhang, J.; Sclaroff, S.; Ikizler-Cinbis, N.; Sigal, L. Space-Time Tree Ensemble for Action Recognition and Localization. *Int. J. Comput. Vis.* **2018**, *126*, 314–332. [\[CrossRef\]](#)
42. Zitnick, C.L.; Dollár, P. Edge Boxes: Locating Object Proposals from Edges. In Proceedings of the Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, 6–12 September 2014; pp. 391–405. [\[CrossRef\]](#)
43. Arbeláez, P.A.; Pont-Tuset, J.; Barron, J.T.; Marqués, F.; Malik, J. Multiscale Combinatorial Grouping. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014), Columbus, OH, USA, 23–28 June 2014; pp. 328–335. [\[CrossRef\]](#)
44. Erhan, D.; Szegedy, C.; Toshev, A.; Anguelov, D. Scalable Object Detection Using Deep Neural Networks. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2014), Columbus, OH, USA, 23–28 June 2014; pp. 2155–2162. [\[CrossRef\]](#)
45. Hosang, J.H.; Benenson, R.; Dollár, P.; Schiele, B. What Makes for Effective Detection Proposals? *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 814–830. [\[CrossRef\]](#)
46. Feng, Y.; Ma, L.; Liu, W.; Luo, J. Spatio-Temporal Video Re-Localization by Warp LSTM. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2019), Long Beach, CA, USA, 16–20 June 2019; pp. 1288–1297.
47. Feng, Y.; Ma, L.; Liu, W.; Zhang, T.; Luo, J. Video Re-localization. In Proceedings of the European Conference on Computer Vision (ECCV 2018), Munich, Germany, 8–14 September 2018; pp. 55–70.
48. Huang, X.; Zhang, J.; Wu, Q.; Fan, L.; Yuan, C. A Coarse-to-Fine Algorithm for Matching and Registration in 3D Cross-Source Point Clouds. *IEEE Trans. Circuits Syst. Video Techn.* **2018**, *28*, 2965–2977. [\[CrossRef\]](#)

49. Zhao, L.; Al-Dubai, A.; Zomaya, A.Y.; Min, G.; Hawban, A.; Li, J. Routing Schemes in Software-defined Vehicular Networks: Design, Open Issues and Challenges. *IEEE Intell. Transp. Syst. Mag (Early Access)*. **2020**. [[CrossRef](#)]
50. Hawbani, A.; Torbosh, E.; Wang, X.; Sincak, P.; Zhao, L.; Al-Dubai, A. Fuzzy based Distributed Protocol for Vehicle to Vehicle Communication. *IEEE Trans. Fuzzy Syst (Early Access)*. **2020**. [[CrossRef](#)]
51. Yeom, S. Multi-Level Segmentation of Infrared Images with Region of Interest Extraction. *Int. J. Fuzzy Log. Intell. Syst.* **2016**, *16*, 246–253. [[CrossRef](#)]
52. Huang, X.; Yuan, C.; Zhang, J. Graph Cuts Stereo Matching Based on Patch-Match and Ground Control Points Constraint. In Proceedings of the Pacific-Rim Conference on Multimedia (PCM 2015), Gwangju, South Korea, 16–18 September 2015; pp. 14–23. [[CrossRef](#)]
53. Huang, X.; Zhang, J.; Fan, L.; Wu, Q.; Yuan, C. A Systematic Approach for Cross-Source Point Cloud Registration by Preserving Macro and Micro Structures. *IEEE Trans. Image Process.* **2017**, *26*, 3261–3276. [[CrossRef](#)]
54. Cai, X.; Shang, J.; Jin, Z.; Liu, F.; Qiang, B.; Xie, W.; Zhao, L. DBGE: Employee Turnover Prediction based on Dynamic Bipartite Graph Embedding. *IEEE Access* **2020**. [[CrossRef](#)]
55. Srivastava, G.; Citulsky, E.; Tilbury, K. The Effects of Ant Colony Optimization on the Anonymization of Graphs. *J. Comput. (JoC)* **2016**, *5*, 92–101.
56. Srivastava, G.; Shumay, M.; Citulsky, E. Social Network Anonymity using Ant Colony Systems. In Proceedings of the International Conference on Computer Games, Multimedia & Allied Technology (CGAT), Singapore, 10–11 April 2017; pp. 64–73.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).