ANALYSIS OF COMPLEX HUMAN INTERACTIONS IN UNCONSTRAINED VIDEOS

BO ZHANG

Advisors: Prof. Dr. **Nicola Conci**

Department of Information Engineering and Computer Science
University of Trento

Italy

March 30th 2015

Final Version

CONTENTS

1

INTRODUCTION

1.1 BACKGROUND

Understanding human activities is a challenging research topic in the community of computer vision, which involves techniques from different fields, for instance, image/video processing, pattern recognition, machine learning, mathematics, psychology, and cognitive science. The interest in behavior analysis has grown dramatically in recent years, due to the increasing societal needs, such as video surveillance [1, 2], event detection [3–5], video summarization [6, 7], and video retrieval [8, 9], to name a few. All these applications require a high-level activity analysis, which consists of multiple atomic actions of individuals.

In fact, human activities can be analyzed at various scales. According to [10], human activities are categorized into four different levels, hierarchically (see Figure 1), namely: gestures, actions, interactions, as well as group activities, which are defined as follows:

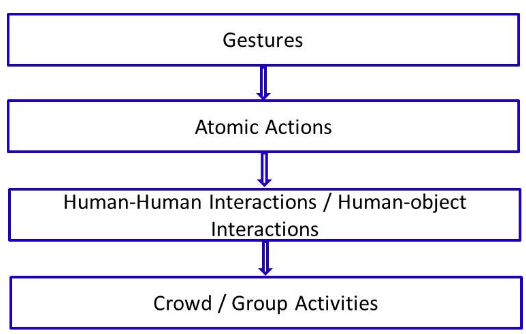


Figure 1: Human activity categorization

- Gestures

Gestures are the atomic motion of body parts characterizing the elementary movements of a person (e.g., *raising an arm*, *waving a hand*, and *stretching a leg*), which are the fundamental elements to compose human actions. Examples [1] of human gestures are shown in Figure 2. Recent works related to gesture recognition include [11–14].

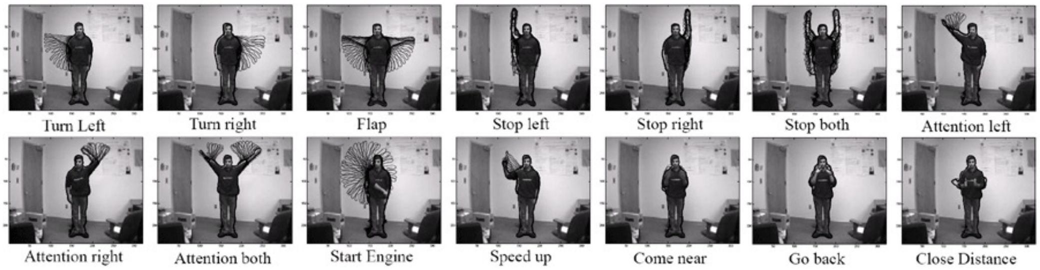


Figure 2: Examples of human gestures.

- Actions

Actions represent single-person activities that are comprised by the concatenations of multiple gestures in specific spatial and temporal orders (e.g., *walking*, *running*, and *jumping*). Recognition of single-person action has already been widely investigated. Examples [2] of atomic human actions are shown in Figure 3.

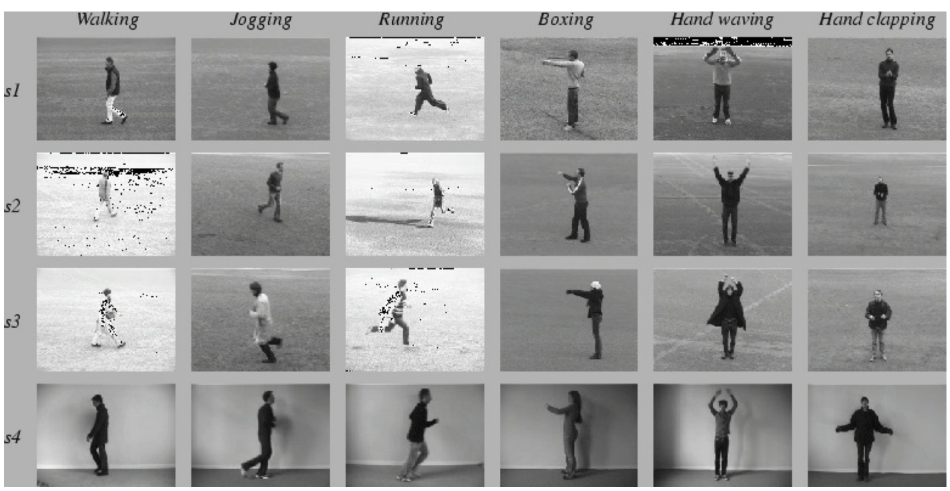


Figure 3: Examples of human actions

- Interactions

Interactions are human activities that require two or more distinctive persons and/or objects, which jointly interpret the event occurring in the scene. Interactions can

1 http://shivvitaladevuni.com/action_rec/action_rec_using_ballistic_dynamics.html
2 http://www.nada.kth.se/cvap/actions/

be further sub-divided into human-human interactions (e.g., *handshake*, *hug*, and *fight*) and human-object interactions (e.g., *cooking in the kitchen*, *stealing luggage*). Examples of human interactions are shown in Figure 4. Related works on human interaction analysis include [15–19].

Figure 4: Examples of human interactions

- Group Activities

Group activities are performed by a large amount of people sharing a common objective, such as *a group marching*, *crossing the road*, and *waiting in a queue*. As the number of people increases, it becomes impossible to isolate each individual's behavior from the crowd (due to background clutters, frequent mutual occlusions, etc.). Examples [3] of group activities are shown in Figure 5. Related works on group analysis include [20–24].

1.2 PROBLEMS AND SOLUTIONS

This thesis focuses the attention on two specific problems related to the interaction analysis, namely discriminative patches segmentation and two-person interaction recognition.

1.2.1 *Problem 1: discriminative patches segmentation*

The motivation behind this topic is that, although videos provide more dynamic information than images, there are still large redundancies between successive frames. Moreover, compared to the whole evolution of human behavior, only a small portion of human activity is essential, and contributes to the classification task. Thus, it becomes important to segment the discriminative patches from video sequences. The potential applications include: video summarization, key-frame extraction, and video matching. Towards this problem, we propose two approaches applied to two

3 `http://wwweb.eecs.umich.edu/vision/activity-dataset.html`

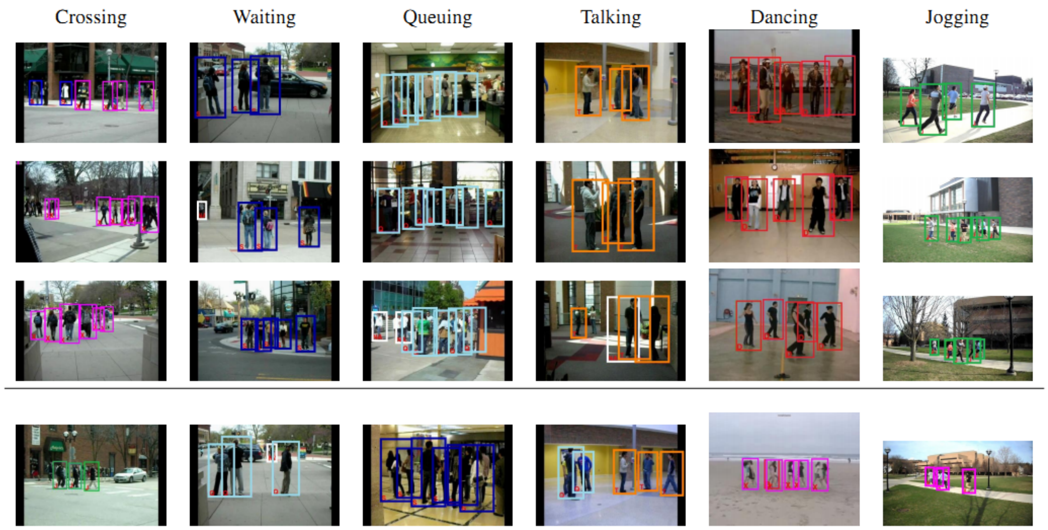


Figure 5: Examples of group activities

different application scenarios: multi-view surveillance cameras and TV shows.

• **Scene 1: Multi-view surveillance cameras**
In this scenario, we collect two-person interaction videos from multi-view surveillance cameras, where people are observed from the lateral view and the bird-eye view, respectively. By exploiting tracking algorithms and proxemics cues from the bird-eye view videos, we infer the interval in which an interaction occurs, and simultaneously segment the corresponding patch from the lateral view video.

• **Scene 2: TV shows**
In this scenario, we exploit the spatio-temporal interest point (STIP) detector to capture the salient motion points in video sequences, and adopt the histogram of oriented gradient (HOG) and histogram of oriented optical flow (HOF) to describe the motion features in the neighborhood of each STIP. Then, we apply the non-negative sparse coding theory and a two-stage *sum-pooling*+l_2-*normalization* scheme to generate better representations of the motion features. Finally, the discriminative patches are extracted using the error-correcting code SVM (ECC-SVM) [25].

1.2.2 *Problem 2: two-person interaction recognition*

The second problem is recognizing complex human interactions in unconstrained videos, which is still an open issue in the computer vision community. With the term '*unconstrained*', we refer to videos collected from movies, TV shows, and surveillance cameras, thus not focusing on specific acquisition strategies. This raises multiple challenges that have to be faced, including: frequent changes of camera viewpoint, fast body movements, and temporal structure modeling, to name a few. With respect to this topic, we propose two different approaches:

• Interaction recognition using the self-similarity matrix (SSM)
In this approach, the motion interchange pattern (MIP) [26] is exploited to detect the abrupt camera viewpoint changes and extract the salient points that are significant to human motions. To deal with fast body movements, we compute the large displacement optical flow (LDOF) [27] on the salient motion points per frame. The temporal correlation of human interactions is modeled using the self-similarity matrix (SSM) on the basis of the histogram of oriented LDOF. After extracting the SSM descriptors, classification is achieved through the standard '*bag-of-words*+SVM' approach.

• Interaction recognition using multiple-instance-learning (MIL) framework
In this approach, we use trajectories to represent low level motion features, and adopt the coherent filtering algorithm [28] to generate the so-called *local motion patterns*. Each *local motion pattern* is described by the histogram of oriented LDOF. Classification is achieved through the multiple-instance-learning (MIL) framework.

1.3 CONTRIBUTION

The main contributions of this thesis are summarized as follows:

- Two different approaches to segment discriminative patches of human interactions from multi-view surveillance cameras and TV shows are proposed. The extracted video clips can preserve the perceptual meaningful portions of human behaviors, and demonstrate better separating capabilities as well (Chapter 3).

- A novel framework for human interaction recognition in TV shows by exploiting the self-similarity matrix (SSM) is presented, where the Motion Interchange Pattern (MIP) is adopted for camera shot boundary detection and salient motion point extraction (Chapter 4).

- A novel framework for human interaction recognition is introduced by analyzing trajectory groups under a multiple-instance-learning perspective (Chapter 5).

1.4 STRUCTURE OF THE THESIS

The rest of the thesis is organized as follows: Chapter 2 reviews the state-of-the-art methods in the field of human activity analysis, in terms of single-action recognition, interaction recognition, as well as the dataset survey. Chapter 3 introduces two *discriminative patch segmentation* approaches, which can be applied in the contexts of multi-view surveillance cameras and TV shows, respectively. In Chapter 4, we propose an effective framework to recognize two-person interactions in TV shows by exploiting the self-similarity matrix (SSM). In Chapter 5, we adopt the multiple-instance-learning (MIL) framework for interaction recognition, which achieves the state-of-the-art performance on the TV human interaction dataset. Conclusions and remarks are discussed in Chapter 6.

2

RELATED WORK

In the last two decades, considerable efforts have been spent by the research community in human activity analysis. A thorough review of the main achievements in this area is presented in [10, 29] and references therein.

2.1 HUMAN ACTION RECOGNITION

For single-person action recognition, local space-time features have been widely adopted. Laptev [30] proposed a spatio-temporal interest point (STIP) detector based on the 3D Harris corner function to capture significant motion patterns from human activities. Dollár *et al.* [31] proposed another interest point detector by applying the Gabor filter in the 3D space, which can detect more interest points compared to Laptev's work. Other interest point detection methods include Hessian detector [32] and dense sampling. As far as the descriptor is concerned, a typical approach is the concatenation of HOG and HOF, which has proved to be an effective descriptor in a wide range of applications. Other interest point descriptors include 3D-HOG [33], extended SURF [32], and MoSIFT [34]. A detailed evaluation of different interest point descriptors can be found in [35]. Differently from interest points, the motion trajectory is another commonly used local space-time feature for motion description. Raptis *et al.* [36] proposed *tracklet* descriptors for video analysis and action recognition. Wang *et al.* [37] proposed dense trajectories to represent motions in a video, where HOG, HOF, and MBH (motion boundary histogram) are used to describe the motion features that surround the trajectory. In [38], they further improved the quality of the trajectory extraction procedure by exploiting the motion compensation. A comparative evaluation of point descriptors and trajectory descriptors can be found in [39]. All these local space-time features are combined with a certain encoding scheme (e.g., *bag-of-words* [40], sparse coding [41, 42], Fisher descriptor [43, 44]), and classification is usually achieved through the standard SVM. More recently, researchers have focused on modeling the global space-time structure for action recognition. In [45], Douglas *et al.* considered activity recognition as a temporal classification problem, and adopted the HMM and CRF for categorization. In [46], Tang *et al.* used the latent structural SVM to model the global temporal structure of human activities, where each type of human activity can be described in terms of different hidden states aligned in the temporal order. In [47], Liang *et al.* modeled human actions through an ensemble of spatio-temporal compositions, and adopted a spatio-temporal AND-OR graph to represent the latent structure of actions.

Another interesting trend that emerges from the recent literatures is the idea of extracting the key and dominant components of human activities. In [48], Satkin *et al.* exploited the *bag-of-words* approach and the standard SVM to determine the essential parts of human activities. In [49], Raptis *et al.* used *poselets* to detect key poses of human activities. In [50], Wang *et al.* proposed a novel model that captures the discriminative motions of human activities, which is known as *motionlets*. In [51], Jain *et al.* proposed a novel activity representation based on mid-level discriminative spatio-temporal patches, and classification is achieved by learning a discriminative SVM classifier. In [52], Sapienza *et al.* adopted the multiple-instance-SVM (MI-SVM) to capture discriminative space-time cuboids related to body movements. In [53], they further incorporated the 3D deformable part model (DPM) into the multiple-instance-learning framework, and developed a more flexible representation of human activities. The advantage of this approach is that it merely exploits the weakly-labeled videos in the training procedure, and all the discriminative body movements are extracted automatically. In [54], Zhu *et al.* developed a mid-level *acton* representation learned through a new max-margin multi-channel multiple-instance learning framework. The learned *actons* are more compact, informative and discriminative in the recognition task. All these approaches are motivated by the need of defining minimal sets of patterns that capture the discriminative portion of an activity.

2.2 HUMAN INTERACTION RECOGNITION

Comparing to the large amount of work on single action recognition, the literature on human interaction recognition is still limited. One possible reason lies in the difficulty of isolating the individuals involved in the interaction, especially when strong mutual occlusions occur. Another non-negligible aspect is the nature of the datasets. Although there are several public interaction datasets available, they are either restricted by the camera viewpoints [55] or are annotated using specific motion capture devices in 3D spaces (i.e., CMU MOCAP dataset [56]). For unconstrained environments, the UT human interaction dataset [57] and the TV human interaction dataset [58] are representative examples collected from surveillance cameras and TV shows, respectively. Moreover, these two datasets are not specifically designed for the recognition task. The UT dataset is created for interaction detection, localization, and categorization, while the TV human interaction dataset is used for interaction retrieval.

In the beginning, hierarchical models were widely adopted. Park and Aggarwal [59, 60] adopted a hierarchical Bayesian network (BN) for two-person interaction recognition, where the segmentation/tracking of different body parts was performed at the low level, and the evolution of poses was estimated using a dynamic Bayesian network (DBN). The recognition of two-person interactions was done by exploiting semantic verbal descriptors (i.e., in 'subject + verb + object' format) at multiple levels: atomic motion at the low level, single-person actions at the middle level, and human interactions at the high level. Ryoo and Aggarwal [61] used a similar hierarchical framework. The main difference lies in the high level, where they proposed a novel representation describing human interactions based on context-free grammars (CFGs) that allow to formally define complex interactions in terms of atomic body movements. Although the above hierarchical models can obtain good classification performance in some specific scenarios, there are still many limitations: (1) they highly rely on the accurate segmentation of human bodies; (2) they require precise trackers; (3) people are always viewed from the lateral viewpoint, namely the best perspective to observe the ongoing interaction. Due to these constraints, it is often unfeasible to apply the hierarchical models into realistic scenarios. In [16], Ryoo and Aggarwal presented a new methodology, which not only allows for human interaction recognition, but can also detect and localize non-periodic activities. Based on the spatio-temporal interest points proposed by Dollár *et al.* [31], they designed a novel matching scheme, known as *spatio-temporal relationship match*, to measure the structural similarities of point features. After introducing the temporal predicates (such as *before*, *after*, *near* and *far*), classification was performed using a hierarchical algorithm. In [62], Marín-Jiménez *et al.* provided a comprehensive analysis on different STIP-based models for interaction recognition, where the dense sampling of STIP proved to be the best strategy. Among the most recent works in interaction analysis, Patron-Perez *et al.* [58] exploited the structured SVM for two-person interaction recognition and localization. The tracking of body parts and the estimation of head poses were achieved at the feature extraction step. Based on the head orientation, local descriptors (HOG features in the local spatio-temporal region around upper bodies) and global descriptors (the relative positions of people) were introduced in their approach. Training and inference were implemented through the structural learning algorithm. This model can tell, which pairs of people are interacting in the scene among several persons, and predict the interaction category and head orientations as well. In [17, 18], Kong *et al.* adopted the so-called *interactive phrases* descriptors to express binary semantic motion relationships between interacting people. These *interactive phrases* descriptors were considered as latent variables, and classification was achieved through the latent structural SVM framework.

2.3 HUMAN INTERACTION DATASET SURVEY

For human activity recognition, a lot of public datasets are available for research purposes, such as: (1) constrained dataset: KTH dataset [40], Weizmann dataset [63], etc. (2) unconstrained dataset: Hollywood2 dataset [64], HMDB [65], etc. More detailed overview of activity datasets can be found in [66]. However, compared to single action recognition, videos for human interaction analysis are very few. In this thesis, we mainly exploit three different human interaction datasets. The details of these datasets are listed below:

• UT Human Interaction Dataset

The UT human interaction dataset [57] is developed by the University of Texas, with the aim of evaluating the algorithms for high-level two-person interaction recognition from surveillance cameras. This dataset consists of 5 different types of human interactions, namely, *handshake*, *puch*, *push*, *kick*, and *hug*.

The videos in this dataset are further divided into two subsets, where SET 1 is recorded on a parking space with almost static background and little camera jitters, while SET 2 is taken on a lawn during a windy day with moving background and more camera jitters. Each type of interaction contains 10 sample videos. Examples of the UT human interaction dataset are shown in Figure 6 .

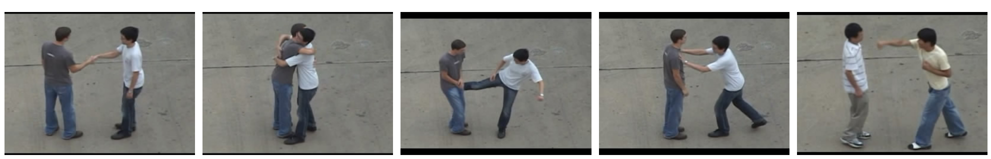


Figure 6: Examples of interactions from the UT human interaction dataset. From left to right: *handshake*, *hug*, *kick*, *push*, and *punch*.

• TV Human Interaction Dataset

The TV human interaction dataset (TVHID) [58] is developed by the visual geometry group (VGG), the University of Oxford, which is used to evaluate algorithms of two-person interaction recognition for retrieval purposes. This dataset consists of four different types of human interactions, such as *handshake*, *highfive*, *hug*, and *kiss*, where each type contains 50 video clips. Moreover, it also has 100 videos that do not contain any interactions, known as *negative samples*. Videos in this dataset are annotated in each frame. The ground truth is comprised by: (1) upper body (person ID, x-coordinate and y-coordinate, scale); (2) discrete head orientation (profile-left, profile-right, frontal-left, frontal-right and backwards); (3) interaction label of each person. This dataset is very challenging due to frequent changes of camera viewpoint, background clutters, multiple people in the scene, and camera motions. Examples of the TVHI dataset are shown in Figure 7.

Figure 7: Examples of interactions from the TV human interaction dataset. From left to right: *handshake*, *highfive*, *hug*, and *kiss*.

• UNITN Social Interaction Dataset

UNITN social interaction dataset (USID) [55] is specifically designed for surveillance and people monitoring. This dataset is recorded outdoor by two different viewpoints: one is from the lateral view, and the other one is from the bird-eye view. For each viewpoint, it includes 4 different types of two-person interactions, namely *handshake*, *hug*, *fight*, and *talk*. Each class contains 16 video clips, with the total number equal to 64 video sequences. All the videos in this dataset show the complete evolution of human interactions. Examples of the USID are shown in Figure 8.

(a) *handshake* (b) *hug* (c) *fight* (d) *talk*

Figure 8: Examples of interactions from the UNITN social interaction dataset. The first row: lateral view. The second row: bird-eye view. From left to right: *handshake*, *hug*, *fight*, and *talk*.

3

DISCRIMINATIVE PATCH SEGMENTATION

3.1 DISCRIMINATIVE PATCH SEGMENTATION FROM MULTI-VIEW SURVEILLANCE CAMERAS

In this section, we first introduce the visual features that exploited in the far-range and close-range analysis, respectively, and then explain how to use the cues from the bird-eye view camera to help the segmentation of discriminative patches from the lateral view camera.

3.1.1 *Far-range Analysis: Proxemics Cues*

Sociological studies have analyzed the behavior of people in a social dimension, defining a set of rules that are generally and unconsciously followed in normal conditions, called *proxemics* [67]. In our work, proxemics cues are gathered from the bird-eye view camera and mapped onto an homographic plane. The retrieved information is then used to trigger the beginning and the end of an interaction, so as to properly identify the subjects involved in the interaction, and simultaneously discard casual or involuntary interactions.

The interaction phase between two subjects is determined using two different energy functions (see Eq. (3.1) and (3.2)): the first one is related to the actual distance between a pair of subjects, and the latter is proportional to the distance between each subject's O-space [55]. The O-space refers to the area immediately in front of the subject, which is commonly considered as the area where the interaction is more likely to happen.

$$E^{d}_{ij} = e^{-\frac{\left\|\kappa^{d}_{ij}\right\|^{2}}{2\sigma^{2}_{w}\sigma^{2}_{d}}} \tag{3.1}$$

$$E^{o}_{ij} = e^{-\frac{\left\|\kappa^{o}_{ij}\right\|^{2}}{2\sigma^{2}_{w}\sigma^{2}_{o}}} \tag{3.2}$$

In (3.1) and (3.2), k_{ij}^{d} is the actual distance between the two subjects, while k_{ij}^{o} indicates the distance between the O-spaces. The parameter σ_w is related to the field of view of the camera: the wider the angle of view, the smaller the interesting interaction area. The terms σ_d and σ_o are two additional parameters related to the proxemics space (e.g., *intimate*, *social*, and *public*) that allows to extend or restrict the interaction area according to the scenario and the position of the camera.

The feature vectors at each frame t contain the energy values over a temporal window τ:

$$E_{i,j}^{d}(t,\tau) = [E_{i,j}^{d}(t-\tau), E_{i,j}^{d}(t-\tau+1), ..., E_{i,j}^{d}(t)] \tag{3.3}$$

$$E_{i,j}^{o}(t,\tau) = [E_{i,j}^{o}(t-\tau), E_{i,j}^{o}(t-\tau+1), ..., E_{i,j}^{o}(t)] \tag{3.4}$$

The feature vectors (3.3) and (3.4) are then transformed into the frequency domain by applying the Fast Fourier Transform (FFT), thus eliminating the temporal correlations. After applying PCA on the above two feature vectors, the obtained values are then concatenated to construct a single feature for each frame. These frame-based features are labeled as *interaction* or *non-interaction*, and then used to train a binary classifier, which can trigger the interaction and provide the temporal interval as well.

The number of available samples N for each video is computed as (3.5):

$$N = \sum_{x=1}^{n} \sum_{y=2, y\neq x}^{n} \mathrm{Frame}(x,y) - \tau \tag{3.5}$$

where n is the number of people in the video, $\mathrm{Frame}(x,y)$ is the number of frames, in which subject x and y are in the scene jointly; τ is the length of the temporal window.

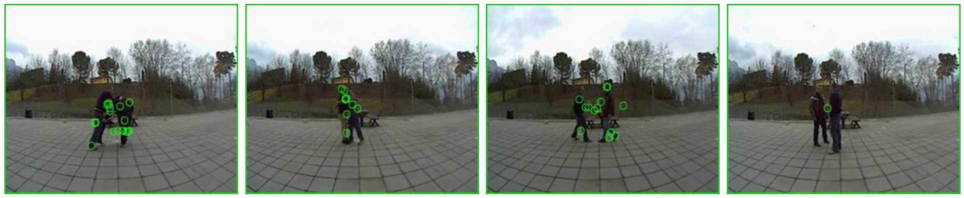
3.1.2 *Close-range Analysis: Spatial-temporal Interest Points*

For close-range analysis, the spatio-temporal interest points are widely adopted for motion representation. In this part, the STIPs are extracted using the 3D Harris corner detector from the lateral view camera. Firstly, we use a function $f(x,y,t)$ to represent a video and compute its linear scale-space representation L by convolution of f with a Gaussian kernel g:

$$L(x,y,t,\sigma_l^2,\tau_l^2) = g(x,y,t,\sigma_l^2,\tau_l^2) * f(x,y,t) \tag{3.6}$$

Then, another Gaussian kernel is adopted to average the 3×3 spatial-temporal matrix composed of first order spatial-temporal derivatives of L:

$$\mu = g(x,y,t,\sigma_i^2,\tau_i^2) * \begin{pmatrix} L_x^2 & L_xL_y & L_xL_t \\ L_xL_y & L_y^2 & L_yL_t \\ L_xL_t & L_yL_t & L_t^2 \end{pmatrix} \tag{3.7}$$



(a) (b) (c) (d)

Figure 9: Examples of STIP detection for different types of two-person interactions in the USI dataset. From left to right: (a) *fight*, (b) *hug*, (c) *handshake* and (d) *talk*.

Next, a 3D Harris corner function H is constructed as:

$$\mathsf{H} = \mathsf{det}(\mu) - k\,\mathsf{trace}^3(\mu) \tag{3.8}$$

We consider the positive local maxima of H as the locations of spatial-temporal interest points. Figure 9 shows four examples of STIP detection on different types of two-person interactions. For each STIP, HOG and HOF are computed in the 3D cuboid around its neighborhood, and represent the spatial and temporal motion features, respectively. The patch is partitioned into a grid with 3×3×2=18 spatial-temporal blocks. Moreover, 4-bin HOG descriptors and 5-bin HOF descriptors are then computed for each block. The feature vector for each detected STIP is represented by the concatenation of both descriptors with the total size of 18×4+18×5=162.

3.1.3 *Patch Segmentation*

The flowchart of temporal interval extraction from the lateral view camera is shown in Figure 10. The main procedure is summarized as follows:

- Step 1: compute the feature vector of each frame discussed in 3.1.1 from the bird-eye view camera;
- Step 2: train a binary SVM classifier as the interaction trigger;
- Step 3: use the trigger to detect and segment the interaction interval from the lateral view camera;

3.1.4 *Evaluation*

In this section, we validate our segmentation approach on the USI dataset. In the far-range analysis, we consider a temporal window τ of 128 frames. The feature

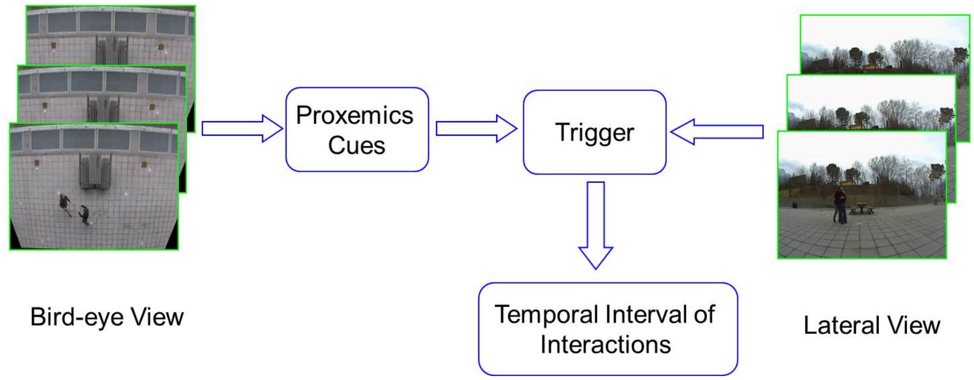


Figure 10: Extracting the temporal interval of two-person interactions from the lateral view camera

vectors corresponding to the distance energy and O-space energy are reduced to 80-dimensionality, respectively. Therefore, the final descriptor for each frame is organized by concatenating the distance feature and O-space feature accordingly, with the total dimensionality equal to 160. The interaction trigger is trained using 5 additional video clips in the USI dataset, which are not adopted for testing. An example of the interaction trigger is shown in Figure 11.

Figure 11: An example of the trigger applied on one *fight* sample, where the white line indicates *absence* of an interaction, and the red line indicates that an interaction is *ongoing*.

In order to demonstrate that the extracted patches are more discriminative and separable, we compare the classification performance on the original videos and the segmented clips, respectively. Considering the limited size of the dataset, we adopt the 64-fold leave-one-out cross-validation strategy. Classification is done using the standard 'STIP + BoW + SVM' scheme, and the corresponding results are listed in Table 1. While Table 2 and 3 further illustrate the confusion matrices with respect to the original videos and the segmented patches.

From Table 1, we can observe that the extracted temporal intervals can improve the classification accuracy by around 3% on average, comparing to the original videos

Table 1: USID: classification results on the original videos and the segmented patches

	# STIPs	# codebook	*fight*	*hug*	*handshake*	*talk*	accuracy
Original Videos	63,161	120	93.75%	81.25%	56.25%	93.75%	81.25 %
Extracted Patches	11,981	80	93.75%	93.75%	68.75%	81.25%	84.375%

Table 2: USID: confusion matrix on the original videos

	fight	*hug*	*handshake*	*talk*
fight	93.75%	0%	0%	0%
hug	6.25%	81.25%	43.75%	6.25%
handshake	0%	18.75%	56.25%	0%
talk	0%	0%	0%	93.75%

from the lateral view directly. The classification results for *hug* and *handshake* are both improved. The decrease accuracy for *talk* is due to the lack of STIPs detected in the sample videos, as in most cases people stand still while talking.

3.2 DISCRIMINATIVE PATCH SEGMENTATION FROM TV SHOWS

In this section, we focus on extracting discriminative video patches in more challenging unconstrained environments, namely, TV shows.

3.2.1 *Non-negative Sparse Coding of STIP Descriptors*

We introduce the sparse coding algorithm used to encode the STIP descriptors. The motivation for choosing a sparse coding strategy is twofold: (i) sparse coding has proven to outperform classical *bag-of-words* schemes in terms of classification accuracy [68]; (ii) sparse coding can also be considered a good tool when dealing with noisy signals (see also [69] and [70]). This second property, in particular, turns out to be quite relevant in our context. In fact, when extracting STIPs, a non-negligible number of points are usually due to camera motions, viewpoint changes, and background clutters. Such points can be regarded as an additional noise source that corrupts the desired activity-related patterns. The adoption of sparse coding helps reducing the impact of such noise, thus allowing to achieve a better representation and consequently a more accurate classification.

In the following paragraphs we briefly summarize the fundamentals of sparse coding and introduce the notations used in the following. Given a dictionary $\mathbf{D} =$

Table 3: USID: confusion matrix on the segmented videos

	fight	*hug*	*handshake*	*talk*
fight	93.75%	0%	0%	6.25%
hug	6.25%	93.75%	31.25%	0%
handshake	0%	6.25%	68.75%	12.50%
talk	0%	0%	0%	81.25%

$[\mathbf{d}_1, \mathbf{d}_2, ..., \mathbf{d}_p] \in R^{m \times p}$, which consists of *p* atoms (each atom $\mathbf{d}_i$ is a column vector), a signal $\mathbf{y} \in R^m$ can be represented in terms of a sparse linear combination of atoms $\mathbf{d}_k$ as follows:

$$\mathbf{y} = \mathbf{D}\boldsymbol{\alpha} + \boldsymbol{e} \tag{3.9}$$

where $\boldsymbol{\alpha}$=$[\alpha_1, \alpha_2, ..., \alpha_p]^T$ is a coefficient vector, and $\boldsymbol{e}$ is the reconstruction error.

The sparse model needs to fulfill two conditions: (i) the number of the non-zero elements in the coefficient vector $\boldsymbol{\alpha}$ is small as compared to p, and (ii) the reconstruction error $\boldsymbol{e}$ is small as compared to $\mathbf{y}$, namely $\|\boldsymbol{e}\| \ll \|\mathbf{y}\|$. The procedure of computing the coefficient vector $\boldsymbol{\alpha}$ is called *sparse coding*, and requires solving an optimization problem, as follows:

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_0 \qquad \text{s.t.} \qquad \frac{1}{2} \| \mathbf{y} - \mathbf{D}\boldsymbol{\alpha} \|_2^2 \leqslant \epsilon \tag{3.10}$$

where $\mathbf{D}$ is an overcomplete dictionary and $\| \cdot \|_0$ is the number of non-zero elements in $\boldsymbol{\alpha}$.

As the optimal solution of Eq. (3.10) is a NP-hard problem, we can approximate it as in Eq. (3.11):

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} \frac{1}{2} \| \mathbf{y} - \mathbf{D}\boldsymbol{\alpha} \|_2^2 + \lambda \| \boldsymbol{\alpha} \|_1 \tag{3.11}$$

where l_0-norm is replaced by l_1-norm, and known as *Lasso* [71].

The procedure of dictionary learning is called *sparse modeling*, and consists on iterating two alternate optimization steps until convergence to a local minimum: (step 1) fixed $\mathbf{D}$ calculate $\boldsymbol{\alpha}$ through Eq. (3.11); (step 2) fixed $\boldsymbol{\alpha}$ update $\mathbf{D}$ through Eq. (3.12):

$$(\mathbf{D}^*, \mathbf{A}^*) = \arg\min_{\mathbf{D},\mathbf{A}} \frac{1}{2} \| \mathbf{Y} - \mathbf{D}\mathbf{A} \|^2 + \lambda \sum_{i=1}^{n} \| \boldsymbol{\alpha}^i \|_1 \tag{3.12}$$

where $\mathbf{A}$=$[\boldsymbol{\alpha}^1, ..., \boldsymbol{\alpha}^n] \in R^{p \times n}$ is a coefficient matrix related to all the n sample signals, and $\boldsymbol{\alpha}^i$=$[\alpha_1^i, ..., \alpha_p^i]^T$ is the coefficient vector of the i-th training sample.

Furthermore, we need to impose that all the elements in $\mathbf{a}$ are non-negative, in order to measure the contribution of each atom in the dictionary. This can be accomplished through the non-negative matrix factorization algorithm.

In our implementation the above technique is applied to STIP descriptors, using the concatenated dictionary. We used *Lasso* to compute the coefficient vector, while we adopted [72] and [73] for online dictionary learning, due to better computational performance on large quantities of high-dimensional data. The procedure can then be defined as follows: given C types of different activities, we train the specific dictionary $\mathbf{D_i}$ for each type of interaction, i=1,2,...,C. Then, the global dictionary $\mathbf{D_{con}}$ is constructed by concatenating all the class-specific dictionaries as:

$$\mathbf{D_{con}} = [\mathbf{D_1}, \mathbf{D_2}, ..., \mathbf{D_C}] \tag{3.13}$$

Finally, we encode the STIP descriptor on the global dictionary, so that joint motion features of different types of activities can be included in the coefficient vectors. Assuming that the length of dictionary $\mathbf{D_i}$ is L_{D_i} (the number of atoms in the dictionary), then the total length of the concatenated dictionary (denoted by L) is obtained as:

$$L = \sum_{i=1}^{C} L_{D_i} \tag{3.14}$$

Thus, each STIP coefficient vector can be expressed by $\mathbf{a_{con}} = [a_1, a_2, ..., a_L]^T$.

3.2.2 *Pooling and Normalization*

After encoding all the STIP descriptors on the concatenated dictionary $\mathbf{D_{con}}$, pooling and normalization procedures are applied to construct the feature vector for each video. In [74], Wang *et al.* provided a comparative study of pooling and normalization methods for action recognition. In our work, a two-stage sum-pooling and l_2-normalization [75] resulted in the configuration returning the best performance.

At the first stage, we do sum-pooling on all the coefficient vectors of STIPs in a video:

$$\Phi = [f_1, f_2, ..., f_L] = \sum_{i=1}^{N} (\mathbf{a}^i_{\mathbf{con}})^{\mathbf{T}} = \sum_{i=1}^{N} [a^i_1, a^i_2, ..., a^i_L] \tag{3.15}$$

where Φ is the video feature vector, N is the number of STIPs detected in a video, $\mathbf{a}^i_{\mathbf{con}}$ is the coefficient vector of the *i-th* STIP descriptor, and L is the length of the concatenated dictionary. Then, we apply the l_2-normalization as follows:

$$\Phi_{l_2} = [f'_1, f'_2, ..., f'_L] = \frac{\Phi}{\sqrt{\sum_{i=1}^{L} {f_i}^2}} \tag{3.16}$$

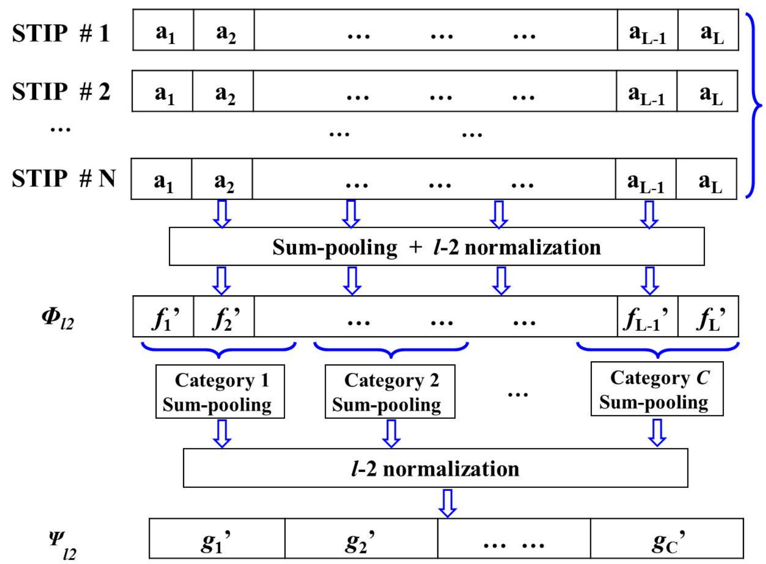


Figure 12: Video feature construction using two-stage sum-pooling and l-2 normalization.

As Φ_{l_2} may not be discriminative enough in complicated scenarios, we do per-class sum-pooling on the video feature vector at the second stage. Assuming that all the class-specific dictionaries have the same length, we then define the second level feature vector Ψ as:

$$\begin{aligned} \Psi &= [g_1, g_2, ..., g_C]; \\ g_k &= \sum_{i=(k-1)*L_{D_k}+1}^{k*L_{D_k}} f_i^{'}; \\ k &= 1, 2, ..., C \end{aligned} \tag{3.17}$$

where $f_i^{'}$ is the component in Φ_{l_2}, L_{D_k} is the length of the class-specific dictionary $\mathbf{D}_k$, and C is the number of activity categories. Again, we normalize Ψ using l_2-norm. The corresponding feature is denoted as Ψ_{l_2}=$[g_1^{'}, g_2^{'}, ..., g_C^{'}]$. The whole procedure of the video feature construction is depicted in Figure 12.

In order to demonstrate the discriminating power of our video feature vector, we compare the classification accuracy obtained on the UT human interaction dataset (set1), using the *bag-of-words* feature and the proposed Ψ_{l_2}, respectively. The UT dataset contains 5 different types of two-person interactions, each of which includes 10 sample videos. The experiment is carried out through a 50-fold *leave-one-out* cross-validation strategy. At the lower level, the STIP descriptor is exploited as the motion feature, while at the higher level, a standard SVM is adopted for classification. The classification results are shown in Table 4 and 5. It can be seen that the proposed two-stage sum-pooling+l_2-normalization strategy provides in this case error-free classification of the dataset, while also reducing the size of the video feature vector to C-dimensionality.

Table 4: Confusion matrix on the UT human interaction dataset (set1) obtained using *bag-of-words* features

	Punch	Hug	Kick	Push	Hand Shake
Punch	50%	0%	40%	0%	10%
Hug	0%	90%	0%	10%	0%
Kick	40%	0%	50%	10%	0%
Push	20%	0%	0%	80%	0%
Hand Shake	0%	0%	0%	10%	90%

Table 5: Confusion matrix on the UT human interaction dataset (set1) obtained using Ψ_{l_2}

	Punch	Hug	Kick	Push	Handshake
Punch	100%	0%	0%	0%	0%
Hug	0%	100%	0%	0%	0%
Kick	0%	0%	100%	0%	0%
Push	0%	0%	0%	100%	0%
Handshake	0%	0%	0%	0%	100%

The UT human interaction dataset is rather simple, as it is recorded in a constrained scenario (fixed camera viewpoint, limited number of persons in the scene, '*staged*' postures). However, we are presenting it here just as a proof of effectiveness of the chosen feature vector Ψ_{l_2}.

3.2.3 *Patch Segmentation*

In order to segment the discriminative patches, we first compute the feature vector for each video Ψ_{l_2}, and then build a multi-class SVM (denoted by w) on the basis of the obtained feature vectors. Next, we determine the category and the corresponding confidence of each patch in a video by exploiting the trained classifier w. For a given video with N frames, we need to compute $N(N\text{-}1)/2$ different patches. Among the patches that can be correctly classified according to the video category, we choose the one with the highest confidence as our candidate.

The confidence of each patch can be expressed by mapping the relevant feature vector x to the posterior probability $p(y|x)$, where y is the video category, and the feature vector x within a patch can be computed in the same way as we did for Ψ_{l_2}. This mapping is achieved through the error-correcting code SVM (ECC-SVM) [25], derived from the generalized Bradley-Terry model.

The ECC-SVM is an efficient tool in dealing with multi-classification problems. Generally, multi-classification is solved by integrating the results from binary classifiers using different strategies including: (1) one-against-one, (2) one-against-all, and (3) error-correcting code. In [76], the authors provided a comprehensive analysis on the classification performance of the above three strategies, where the error-correcting code scheme demonstrates better classification capabilities.

Error-correcting code is a general framework that aims at enhancing the generalization ability of binary classifiers. It decomposes a multi-class problem into several binary classification tasks, and combines the results of these base classifiers (e.g., SVMs, naive Bayes). In this paper, we adopt the ECC-SVM proposed in [25] to generate the multi-class prediction and probability estimates. SVMs with RBF kernel are considered as the base binary classifiers. On top of that, '*one-against-the rest*' strategy is used as the ECC encoding scheme, as it is competitive against the other three schemes (namely, *one-against-one*, *dense*, and *sparse*), while also being computationally efficient. For each input x (patch feature vector), the confidence (posterior) for each class is obtained by solving the Generalized Bradley-Terry models. We then choose the class that has the highest posterior as the prediction of the patch. More details about the optimization of the Generalized Bradley-Terry models can be found in [25].

The segmentation of the most discriminative patch is then computed for a given video i, as reported in Eq. (3.18):

$$\begin{aligned}[\mathrm{frame}^*_{\mathrm{start}}, \mathrm{frame}^*_{\mathrm{end}}] = \\ \arg \max_{\forall:\mathrm{start}\leqslant\mathrm{end}} \{\mathbf{conf}(w, \mathbf{video}_i(\mathrm{frame}_{\mathrm{start}}, \mathrm{frame}_{\mathrm{end}}), c_i)\}\end{aligned} \quad (3.18)$$

where, c_i corresponds to the video category for the video i, and $\mathrm{frame}^*_{\mathrm{start}}$ and $\mathrm{frame}^*_{\mathrm{end}}$ indicate the position of the discriminative patch. The function $\mathbf{video}()$ computes the feature Ψ_{l_2} within the range indicated by $\mathrm{frame}_{\mathrm{start}}$ and $\mathrm{frame}_{\mathrm{end}}$, where we assume $\mathrm{start}\leqslant\mathrm{end}$. Given the classifier $\boldsymbol{w}$ learned from the activity videos, the function $\mathbf{conf}()$ computes the posterior of a patch that can be classified as c_i. The patch that has the highest confidence is considered to be the most discriminative portion. A more detailed description about the computation of function $\mathbf{conf}()$ is shown in Algorithm 1.

Algorithm 1: CONFIDENCE

Input: w, the $\mathtt{i\text{-}th}$ video, $\mathtt{frame}_{\mathtt{start}}$ and $\mathtt{frame}_{\mathtt{end}}$, $\mathtt{c_i}$
Output: confidence (posterior) of the patch
Step 1: Compute the feature vector Ψ_{l_2} of a video patch indicated by $\mathtt{frame}_{\mathtt{start}}$ and $\mathtt{frame}_{\mathtt{end}}$;
Step 2: Classify the video patch using the given classifier w, the prediction is denoted as $\mathtt{class}$;
Step 3:
if *(*$\mathtt{class} == \mathtt{c_i}$*)* **then**
    return the corresponding confidence;
else
    return NULL;
end

In Figure 13, we show some examples of the patch extraction procedure. We display four different types of two-person interactions in the first column, selected from the TVHI dataset: *handshake* (15*th* video in the dataset), *highfive* (33*rd* video), *hug* (3*rd* video), and *kiss* (37*th* video). The second column reports the confidence map obtained by the ECC-SVM. Each pixel in the map represents the confidence of a specific patch, where the y-coordinate indicates the start frame, and the x-coordinate indicates the end frame. The blank space represents the invalid patches, namely where $\mathtt{frame}_{\mathtt{start}} > \mathtt{frame}_{\mathtt{end}}$ or in case no STIP can be detected. The diagonal specifies the confidence of each frame. The third column refers to the prediction map of the video patches. Only the colored portion of the map represents the patches that are correctly classified in accordance to their categories. The prediction map can be considered as a mask. We apply the mask on the confidence map using a logical '*and*' operation and generate the final results (see the fourth column), where patches with the highest confidence are selected as the candidates. The last column shows the representative frames within the discriminative patches.

3.2.4 *Evaluation*

In this section we will detail the procedures used for validation and the corresponding results. At first we demonstrate the discriminating capability of the feature vector Ψ_{l_2} on the TVHI dataset. Specifically, we highlight the contribution of pooling and normalization at each step in the feature construction procedure (see Figure 12) with respect to the final classification performance. Then, we verify that the patches extracted by our approach can preserve the essential semantic meaning of human activities by exploiting the ground truth of the TVHI dataset. Finally, we show that

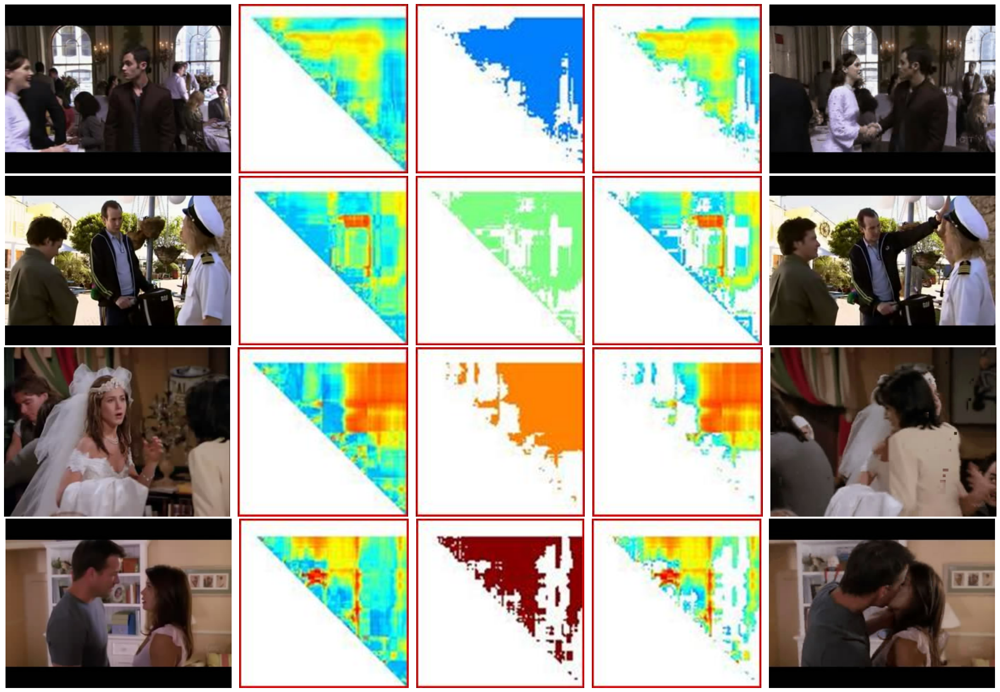


Figure 13: The patch extraction procedures of four different types of human interactions : *handshake*, *highfive*, *hug* and *kiss*. From the 1st column to the 5th column: the *original video* (the first frame), *confidence map*, *category mask*, the *results after 'and' operation*, and the corresponding *representative frames* in the patches. The sample videos are selected from the TVHI dataset.

the segmented patches are more separable than the entire videos, by validating on the TVHI and Olympic sports datasets.

3.2.4.1 *TVHI Dataset*

This dataset consists of four different interactions: *handshake*, *highfive*, *hug* and *kiss*. Each category of interactions contains 50 sample videos. The dataset is very challenging due to a high degree of variability between videos, in terms of number of persons in the scene, camera viewing angle, and shot changes.

First, we extract all the STIPs from the dataset, with a total number of STIPs equal to 246,619. Each STIP is represented by the concatenation of the HOG and HOF features, for an overall dimensionality equal to 162. Then, class-specific dictionaries are learned on the STIP descriptors. In order to guarantee the sparsity requirements, we set the size of each class-specific dictionary to 1,000, for a total size of the concatenated dictionary equal to $4\times1{,}000{=}4{,}000$. For dictionary learning, the number of iterations is set to 1,000. The non-negative matrix factorization and sparse coding procedure are implemented using the tool SPAMS 2.3 [77].

According to Eq. (3.9), we define the relative reconstruction error as $\|e\|/\|y\|$ in l_2-norm. The sparsity of each STIP coefficient vector is calculated as the value $\|\alpha\|_0$ divided by the size of the concatenated dictionary. The corresponding value represents therefore the percentage of non-zero elements in the coefficient vector. The average relative reconstruction error and the sparsity of STIPs of different interaction types are listed in Table 6. From Table 6, we can conclude that all the STIPs can be reconstructed accurately by their class-specific dictionaries, while also keeping the coefficient vectors sparse.

Table 6: Average relative reconstruction error and sparsity of STIPs from different interaction types

	STIP Number	Error	Sparsity
handshake	60,653	2.5895%	12.63%
highfive	36,384	1.8586%	13.14%
hug	98,841	2.7060%	12.44%
kiss	50,741	2.4009%	12.61%

• **The discriminating capability of different video feature vectors**

In this paragraph, we demonstrate the discriminating capability of the feature vector at each step (see Figure 12), and emphasize the effect of per-class sum-pooling operation, as well as the importance of l_2-normalization. The discriminating power of the features is evaluated in terms of the classification accuracy using the 200-fold *leave-one-out* cross-validation strategy. For comparison, we also provide the classification accuracy using the standard '*bag-of-words*+SVM' approach as the baseline. The same strategy is used for evaluation. We first apply the *k-means* clustering on STIP descriptors to generate the so-called visual codebook. The size of the visual codebook ranges from 400 to 600, in steps of 50. An RBF kernel is adopted for the SVM, where γ and C are fixed, and obtained through the grid search. Additionally, we further compare the performance of the STIP-based model presented in [78], which exploits different feature selection criteria, such as *information gain* (IG) and *knowledge gain* (KG), to 'discover' more discriminative visual words from the codebook. The details of the procedure followed to carry out the comparison is reported hereafter:

1. STIP + *bag-of-words* model (see Table 7)
2. Classification using different feature selection from the visual codebook [78] (see Table 8)

3. Classification using Φ_{l_2} (see Table 9)
4. Classification using Ψ (without l_2-normalization, in order to show the effect of per-class sum-pooling, see Table 10)
5. Classification using Ψ_{l_2} (in order to highlight the importance of l_2-normalization, see Table 11)

Additionally, we further explain the feature selection strategies adopted in [78]: since STIPs may be extracted also in areas of the video that are not representative of the interaction (e.g., due to noise in the background, shadows, compression artifacts), it is necessary to identify the most significant visual words contained in the codebook. To this aim, *knowledge gain* (KG) is proposed as a solution to evaluate the importance of each visual word based on the *rough set* theory. According to the *rough set* theory [79], *knowledge* is considered as an ability to partition objects on their properties, defined as *knowledge quantity*. Here, we introduce several relevant definitions below:

Definition 1 Knowledge quantity - The object domain U is divided into m equivalence classes by a set of features P. The probability of elements in each equivalence class is $p_1, p_2, ..., p_m$. Let W_p denote the knowledge quantity of P as $W_p = W(p_1, p_2, ..., p_m)$, which satisfies the following conditions:
(1) if $m = 1$, $W_p = 0$;
(2) $W(p_1, ..., p_i, ..., p_j, ..., p_{m)} = W(p_1, ..., p_j, ..., p_i, ..., p_{m)}$;
(3) $W(p_1, p_2, ..., p_{m)} = W(p_1, p_2 + ... + p_m) + W(p_2, ..., p_{m)}$;
(4) $W(p_k, p_m + p_n) = W(p_k, p_m) + W(p_k, p_n)$;

Definition 2 Conditional knowledge quantity - Given U the object domain, P and D the two feature sets, and v_j a specific value of P, then the conditional knowledge quantity $W_{D|P}$ is defined as:

$$W_{D|P} = \sum_j \mathrm{prob}(P = v_j) W_{D|P=v_j} \tag{3.19}$$

Definition 3 Knowledge gain - From Eq. (3.19), the knowledge gain $KG(D \mid P)$ is defined as:

$$KG(D \mid P) = W_{D|P} - W_D \tag{3.20}$$

The knowledge gain measures the amount of knowledge obtained for category prediction by knowing the presence or absence of an item in the feature set.

Considering m as the equivalence class, and that the cardinality of each equivalent class is $n_1, n_2, ..., n_m$, then the knowledge quantity of P can be obtained by as:

$$W_p = W(p_1, p_2, ..., p_m) = \sum_{1 \leqslant i < j \leqslant m} p_i \times p_j \tag{3.21}$$

where $p_i = n_i/(\sum_{i=1}^{m} n_i)$. Given two sets $C = \{c_1, c_2, ..., c_m\}$ and $T = \{t_1, t_2, ..., t_k\}$, where C represents the categories of human interactions and T represents the visual words, the knowledge gain of the visual word t can be computed by Eq. (3.22), where $p(c_i|t)$ and $p(c_i|\bar{t})$ indicate the information obtained for category prediction by knowing the presence/absence of the visual word t, respectively.

$$\begin{aligned} KG(t) = KG(C \mid t) = W_{C|t} - W_C = -\sum_{1 \leqslant i \leqslant j \leqslant m} p(c_i)p(c_j) \\ + [p(t) \sum_{1 \leqslant i \leqslant j \leqslant m} p(c_i|t) \log p(c_i|t) + p(\bar{t}) \sum_{1 \leqslant i \leqslant j \leqslant m} p(c_i|\bar{t}) \log p(c_i|\bar{t})] \end{aligned} \tag{3.22}$$

The knowledge gain can measure the importance of each visual word in the codebook. After sorting all the visual words by descending order according to their knowledge gain values, we select the items, for which the percentage of the accumulated knowledge gain is larger than a predefined threshold. The selected visual words will be used to construct a new feature vector for each video. These new feature vectors are used to train a multi-class SVM for interaction classification. As a baseline, another typical method, known as *information gain* is also adopted in selecting the discriminative visual words from the visual codebook. More details about *information gain* can be found in [80].

Table 7: Confusion matrix: STIP + *bag-of-words* model, size of codebook=550, average accuracy=45.5%

	handshake	highfive	hug	kiss
handshake	36%	16%	22%	26%
highfive	22%	42%	14%	22%
hug	12%	10%	68%	10%
kiss	20%	18%	26%	36%

From Table 9 and Table 10, it can be seen that the average accuracy can be improved by the per-class sum-pooling operation (from 33.5% to 46%). From Table 10 and Table 11, it can be noticed how the l_2-normalization can further improve the classification performance (from 46% to 64%). Comparing these figures with the baseline (see Table 7), we can conclude that Ψ_{l_2} consists of a better representation of the videos in this dataset.

As we mentioned in the previous sections, the use of dense trajectories has demonstrated strong capabilities in performing activity recognition [37], [38]. Therefore,

Table 8: Classification accuracy using feature selection from the visual codebook [78]

	hand-shake	highfive	hug	kiss	Average
Baseline	36%	42%	68%	36%	45.5%
IG	38%	50%	68%	36%	48%
KG	46%	50%	58%	48%	50.5%

Table 9: Confusion matrix: Φ_{l_2}, dim=4000, average accuracy=33.5%

	handshake	highfive	hug	kiss
handshale	18%	52%	18%	12%
highfive	10%	46%	12%	32%
hug	22%	20%	44%	14%
kiss	16%	46%	12%	26%

Table 10: Confusion matrix: Ψ (without l_2-normalization), dim=4, average accuracy=46%

	handshake	highfive	hug	kiss
handshake	42%	26%	16%	16%
highfive	24%	42%	14%	20%
hug	20%	18%	46%	16%
kiss	16%	14%	16%	54%

Table 11: Confusion matrix: Ψ_{l_2}, dim=4, average accuracy=64%

	handshake	highfive	hug	kiss
handshake	74%	6%	4%	16%
highfive	20%	56%	10%	14%
hug	26%	16%	46%	12%
kiss	6%	10%	4%	80%

we also compare our feature representation (Ψ_{l_2}) against dense trajectories, adopting MBH as the motion feature for each trajectory. Classification performance is evaluated using the same criterion as mentioned above. The obtained classification results are reported in Table 12-14, using different sizes of the codebook (50, 100, and 500, accordingly). Also in this case we can confirm the suitability of our feature representation.

Table 12: Confusion matrix: dense trajectory + *bag-of-words* model, size of codebook=50, average accuracy=53%

	handshake	highfive	hug	kiss
handshake	32%	16%	30%	22%
highfive	10%	72%	14%	4%
hug	8%	2%	74%	16%
kiss	12%	6%	48%	34%

Table 13: Confusion matrix: dense trajectory + *bag-of-words* model, size of codebook=100, average accuracy=56.5%

	handshake	highfive	hug	kiss
handshake	38%	10%	38%	14%
highfive	8%	72%	16%	4%
hug	0%	4%	84%	12%
kiss	16%	6%	46%	32%

Table 14: Confusion matrix: dense trajectory + *bag-of-words* model, size of codebook=500, average accuracy=53.5%

	handshake	highfive	hug	kiss
handshake	38%	14%	36%	12%
highfive	6%	74%	10%	10%
hug	2%	4%	76%	18%
kiss	8%	12%	54%	26%

• **Perceptual relevance of the extracted patches**

In this section, we first demonstrate that the discriminative patches extracted by our approach can preserve the core part of human activities, i.e., the portion of the video that contains the most significant pattern of the action from a perceptual point of view. Then, we show that the patches are more separable compared to the original videos.

After extracting the patches, whose length is clearly shorter than the original video (please refer to Table 15), we aim at verifying that the obtained patch is accurate enough in representing the corresponding human activity. To do so, and to provide a quantitative analysis, we exploit the ground truth file of the TVHI dataset. The ground truth provides the temporal interval for every ongoing interaction. We then compare the patches with the ground truth for each video in terms of the precision rate and recall rate, defined as follows:

Table 15: Average length (in frame) of different types of discriminative patches

Average Length	handshake	highfive	hug	kiss
Patch	14.72	6.48	12.62	15.52
Original Video	77.48	48	120.66	100.30

$$\begin{aligned} \mathrm{precision} &= \frac{N_{\mathrm{interaction}}}{L_{\mathrm{patch}}} \\ \mathrm{recall} &= \frac{N_{\mathrm{interaction}}}{L_{\mathrm{groundtruth}}} \end{aligned} \tag{3.23}$$

where $N_{\mathrm{interaction}}$ is the total number of frames labeled as 'interaction' within the patch. L_{patch} represents the length of the patch, and $L_{\mathrm{groundtruth}}$ indicates the duration of the interaction provided by the ground truth. A detailed illustration is presented in Figure 14.

We compute the average precision and recall rates for each interaction category, and the results are shown in Table 16. From Table 16, we can find that, on average, around half of the frames (51.47%) in the patches fall within the interaction interval, while these frames only occupy a small portion (around 21.64%) of the whole activity.

For completeness, we report some sample images from different interaction patches in Figure 15. These patches are segmented automatically from the first 8 sample videos of each type, and the central frame of the patch is displayed. From Figure 15, we can see that most of the video patches can capture the significant elements of the corresponding human interaction, by only using nearly 15% of the original length (see Table 15). As expected, the patches within the same category exhibit similar motion patterns, while the irrelevant portions of the original videos are filtered out.

We present now two other examples of the discriminative patches in Figure 16 and 17, respectively. We have selected the 10-*th handshake* video and 12-*th hug* video in

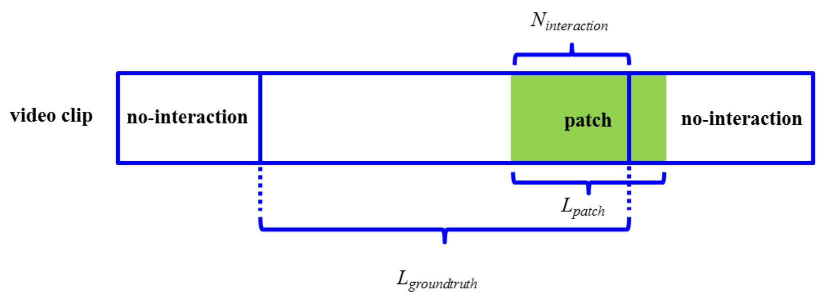


Figure 14: Illustration of precision/recall rate for a given video clip. The green rectangle represents the discriminative patch extracted by our approach. $\mathtt{L_{groundtruth}}$ indicates the period that a certain interaction is ongoing. The remaining frames are labeled as 'no-interaction'.

Table 16: Average precision rate (AP) and recall rate (AR) of the patches with respect to each category.

	AP	AR
handshake	51.38%	19.99%
highfive	35.31%	24.25%
hug	58.95%	16.42%
kiss	60.24%	25.90%
Average	51.47%	21.64%

Figure 15: Examples of the discriminative patches extracted by our approach. From the top row to the bottom: *handshake*, *highfive*, *hug*, and *kiss*. For conciseness only the central frame of the patch is displayed.

the TVHI dataset. The discriminative portions are highlighted using blue bounding boxes. We can notice how the frames that do not contain relevant information for the activity of interest are discarded, thus limiting the duration of the patch to only a small portion of the original videos, while still preserving the distinct motion patterns of the corresponding activities.

In order to verify that the extracted patches are more separable compared to the original videos, we repeat the 200-fold *leave-one-out* cross-validation strategy on these patches. The results are shown in Table 17. Comparing with the figures presented in Table 11, it is possible to appreciate the better separability of the patches.

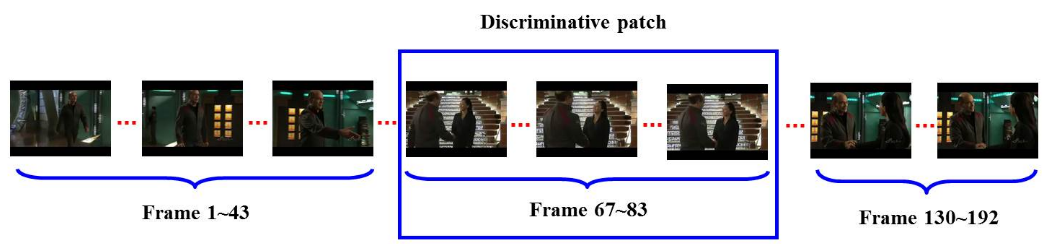


Figure 16: An example of the discriminative patch segmented from the 10-*th handshake* video in the TVHI dataset.

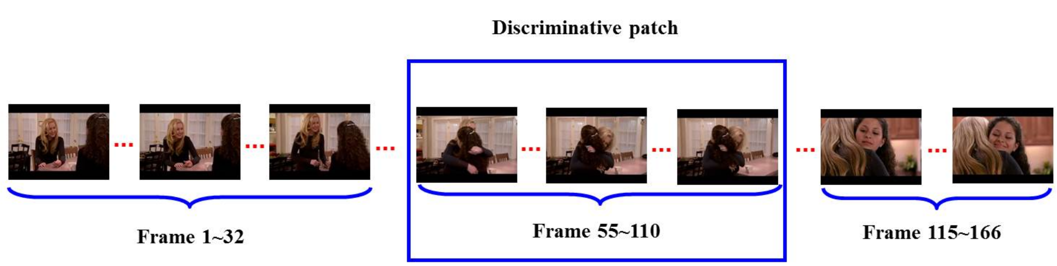


Figure 17: An example of the discriminative patch segmented from the 12-*th hug* video in the TVHI dataset.

It is worth noting that the proposed method focuses on the segmentation of the most significant temporal frames of a given action. We demonstrate that we are

Table 17: Confusion matrix: classification results on the extracted patches, average accuracy=68.5%

	handshake	highfive	hug	kiss
handshake	64%	16%	2%	18%
highfive	10%	58%	20%	12%
hug	16%	4%	74%	6%
kiss	12%	10%	0%	78%

able to do this with good results in terms of precision rate. Notwithstanding this, sometimes the segmented pattern does not contain the entire action, as can be noticed by the value of the recall rate. In fact, in most cases, this does not impact negatively on the classification accuracy, which instead greatly benefits of the patch extraction. However, for a small number of samples, this is not true. One of these specific cases is the *handshake*. Observing this class we can infer that the drop in performance is due to the similarity of this action with the motion patterns of *highfive*, where most of the misclassified samples are transferred.

3.2.4.2 *Olympic Sports Dataset*

In this section, we validate our method on another challenging context, namely the Olympic sports dataset [81]. This dataset includes 16 different categories of sports. We extract the discriminative patches from the training set, which contains 650 videos in total. We set the dimensionality of each class-specific dictionary to 1,000, leading to a total size of the concatenated dictionary equal to 16,000.

As the average duration of the videos in this dataset is longer compared to the TVHI dataset, we add the constraint that the minimum duration of the patch should be at least 30 frames, in order to incorporate enough motion information. The average lengths of the original videos and the discriminative patches are reported in Table 18.

We compute the classification accuracy on the original videos and the corresponding patches through a 650-fold *leave-one-out* cross-validation strategy. The corresponding classification results are shown in Table 19.

The results in Table 18 confirm that also in this case the average length of the discriminative patches (68.52 frames) is only 18.4% of the original video length. As shown in Table 19, the classification performance in most of the sport categories increases, and for some classes it improves significantly. Only the accuracy of *javelin-throw* and *shot-put* reports a decrease.

Table 18: Olympic Sports Dataset: The average length of the original videos and patches (in frame)

Sport Type	Original Videoes	Patches
1. basketball-layup	193.65	55.15
2. bowling	335.85	76.44
3. clean-and-jerk	820.5	82.16
4. discus-throw	312.93	50.88
5. diving-platfrom-10m	339.71	53.79
6. diving-springboard-3m	397.53	85.05
7. hammer-throw	416.82	49.08
8. high-jump	292.05	95.71
9. javelin-throw	256.10	74.76
10. long-jump	360.48	61.03
11. pole-vault	358.09	107.97
12. shot-put	271.30	45.15
13. snatch	485.10	48.03
14. tennis-serve	420.13	77.31
15. triple-jump	496.24	64.53
16. vault	209.02	69.26
Average	372.84	68.52

Table 19: Olympic Sports Dataset: classification results on the original videos and the patches.

Sport Type	Original Videoes	Patches
1. basketball-layup	92.50%	**97.5%**
2. bowling	97.56%	**100%**
3. clean-and-jerk	100%	100%
4. discus-throw	90.38%	**94.23%**
5. diving-platfrom-10m	97.92%	**100%**
6. diving-springboard-3m	84.21%	**97.37%**
7. hammer-throw	89.47%	**100%**
8. high-jump	80.35%	**94.64%**
9. javelin-throw	90.48%	85.71%
10. long-jump	92.50%	**95%**
11. pole-vault	68.75%	**96.88%**
12. shot-put	96.23%	94.34%
13. snatch	100%	100%
14. tennis-serve	87.50%	87.50%
15. triple-jump	76.47%	**88.24%**
16. vault	93.47%	**97.83%**
Average Accuracy	89.68%	**95.58%**

4

INTERACTION RECOGNITION USING SELF-SIMILARITY MATRIX

In this chapter, we focus on recognizing complex human interactions in TV shows by exploiting the self-similarity matrix (SSM). The challenging problems we need to deal with in this scenario include: (1) frequent changes of camera viewpoint; (2) multiple people moving in the scene; (3) fast body movements, and (4) videos of short durations that often do not capture sufficient contextual information.

In our framework, we exploit the motion interchange pattern (MIP) [26] to detect the abrupt changes of camera viewpoint, so as to filter out the frames that hinder the feature extraction procedure. For multiple people moving in the scene, we use the bounding boxes around human upper bodies to identify persons who are involved in the interactions. In order to deal with fast body movements, we adopt the large-displacement optical flow (LDOF) [27] to estimate the motion information for each pixel (i.e., velocity, direction). Since the traditional tools (*e.g.*, HMM, CRF) are not suitable for modeling the temporal structure of human activities in very short clips, we use the self-similarity matrix (SSM) based on the histogram of oriented LDOF per frame to model the temporal correlation of human interactions. Moreover, we focus on the region of interest (ROI) that covers the interacting people in each frame, with the purpose of highlighting the role of motion features in the classification task. Examples of the ROI are presented in Figure 18.

Figure 18: Examples of ROI. The green bounding boxes indicate human upper bodies, while the red bounding boxes are the interest regions. All the ROI are provided by [58].

The overall procedure of our approach is presented in Figure 19, and the details are further discussed in the following paragraphs.

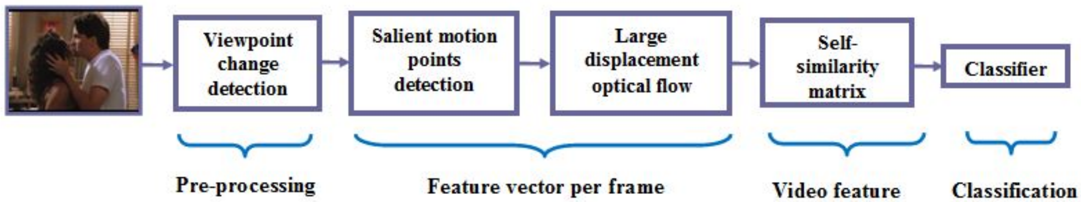


Figure 19: The proposed framework.

4.1 SHOT BOUNDARY DETECTION

Videos in TV shows always contain frequent changes of camera viewpoint, known as *shot boundary*. These boundaries exert side-effects in the feature extraction procedure. In this part, we introduce a novel approach that exploits the motion interchange pattern (MIP) for shot boundary detection.

4.1.1 *Motion interchange pattern*

The MIP is first used for measuring the similarity between two activity videos. For each pixel $p(x, y, t)$ in a video, the MIP encodes the pixel using 8 strings, each of which consists of 8 trinary bits. Thus, the MIP descriptor of pixel p, denoted by $S(p)$, is comprised by 8×8=64 bits.

For each bit in $S(p)$, the encoding scheme computes the compatibilities of a local 3×3 patch centered at p in the current frame with respect to two different patches (denoted as i and j, respectively) located in the previous and next frames. The eight possible locations of patches in each of the previous and next frame are shown in Figure 20. The center of the patch in the current frame is denoted as (0,0). The eight possible locations corresponding to the central pixel location in each of the previous and next frame can be defined as: (4,0), (-3,3), (0,4), (3,3), (4,0), (3,-3), (0,-4), and (-3,-3). The angle between patch i and j is denoted as $\alpha = 0°$, 45°, 90°,..., 315°. When considering all the combinations of i and j, the encoding scheme generates a 64-bit descriptor $S(p)$, each bit of which, denoted as $S_{i,j}(p)$, corresponds to a different combination of i and j.

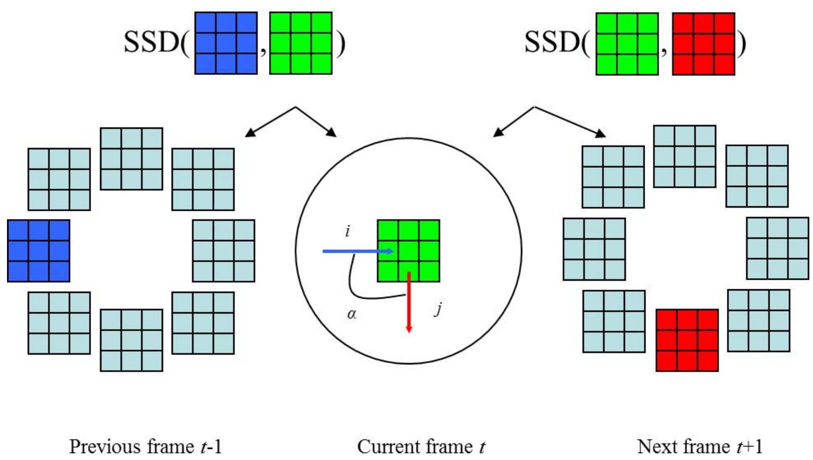


Figure 20: An example of the motion interchange pattern.

The compatibility of two local patches is measured through the sum of squared differences (SSD) score. Each $S_{i,j}(p)$ is computed according to Eq. (4.1), where the threshold θ is set to 1296 empirically according to [26].

$$S_{i,j}(p) = \begin{cases} +1; & \text{if } SSD_i - SSD_j > \theta \\ 0; & \text{if } |SSD_j - SSD_i| \leqslant \theta \\ -1; & \text{if } SSD_i - SSD_j < -\theta \end{cases} \tag{4.1}$$

A value of '-1' indicates that the previous patch i is more similar to the patch in the current frame, while '+1' indicates that the patch j in the next frame is more likely to be the candidate. A value of '0' indicates that both are compatible.

4.1.2 *One-class SVM for shot boundary detection*

In our work, we only consider the direction α=0 ° when computing the MIP for each pixel, thus generating a 8-bit indicator, which is similar to the local trinary pattern (LTP). The indicator consists of '-1', '0', and '+1' bits. We separate the indicator into the positive portion (comprised by the '+1' bits) and the negative portion (comprised by the '-1' bits). We compute the number of non-zero bits for each pixel, and then build a 9-dimensional histogram with the range of [0,8] for each frame. On the other hand, we subtract the number of '+1' bits and the number of '-1' bits for each pixel, and generate a 17-dimensional histogram for each frame, ranging from -8 to 8. The concatenation of these two histograms is considered as the feature vector for each frame, which is fed to the shot boundary detector in the next step.

We category the video frames into 3 different types, namely the regular frame, the end of the previous viewpoint, and the start of the next viewpoint, which are denoted as *regular-frame*, *end-frame*, and *start-frame*, accordingly. The *end-frame* and *start-frame* have distinct visual patterns in the MIP map, while the frames within the same viewpoint do not exhibit a evident pattern (see Figure 21 and 22, respectively). In Figure 21, the 1st and 3rd rows demonstrate two shot boundaries in the video sequences. The 2nd and 4th rows show the corresponding MIP feature map for each frame. The blue-color frame corresponds to the *end-frame*, while the red-color frame corresponds to the *start-frame*. The shot boundary consists of one *end-frame* and one *start-frame* aligned in the temporal order. Figure 22 demonstrates the regular frames within a video, which does not contain any shot changes.

Figure 21: The typical pattern of the shot boundary.

As the shot boundary frames only occupy a small portion of the video length, we use the one-class SVM for detection. The one-class SVM [82, 83] is a type of classifier, which aims at describing the distribution of data from a specific category. It is widely adopted in the case when the training samples are unbalanced, like *outlier detection*, *fault diagnosis*, etc.

In our work, we adopt the one-class SVM proposed by [83], known as SVDD. This algorithm creates a spherical boundary that surrounds the data in the feature

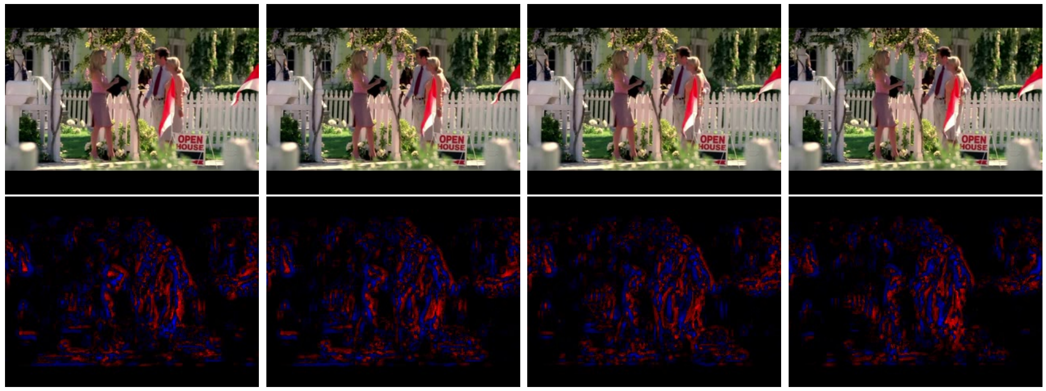


Figure 22: Examples of regular frames. The clips are collectd from a scenario with a static background.

space by minimizing the volume of the hypersphere. The resulting hypersphere is determined by the center *a* and the radius *R*, where *a* is a linear combination of the support vectors, and *R* indicates the distance from the center to the boundary. The volume of the hypersphere can be measured in R^2. In order to deal with the problem of over-fitting, slack variables ξ_i and the penalty term C are introduced to create a soft margin, which endures that some distances from data points x_i to the center a can be larger than *R*. The optimization formulation is summarized as:

$$
\begin{aligned}
&\min_{R,a,\xi_i} \|R^2\| + C\sum_{i=1}^{n} \xi_i \\
&\text{s.t.} \quad i = 1, 2, ..., n \\
&\|x_i - a\|^2 \leqslant R^2 + \xi_i \\
&\xi_i \geqslant 0
\end{aligned}
\tag{4.2}
$$

The formulation can be solved using the dual format by introducing the Lagrange multipliers and Gaussian kernel function. For each sample *z* in the test set, it can be accepted when:

$$
\sum_{i=1}^{n} \alpha_i exp(\frac{-\|z - x_i\|^2}{\sigma^2}) \geqslant -\frac{1}{2}R^2 + C_R \tag{4.3}
$$

where x_i is the support vector, α_i is the Lagrange multipliers, C_R only depends on the support vectors. More details about the SVDD algorithm can be found in [83].

In our application, we design two types of one-class SVM, C_1 and C_2, corresponding to the *end-frame* and *start-frame*, respectively. Most of the regular frames can be viewed as *outliers*. The indicator of each frame i in a video (denoted as

$\mathtt{indicator}[i]$) is computed as in Eq. (4.4). The salient pattern of the shot boundary is detected as two successive indicators (namely '-1' and '+1') appeared in a video sequence. Figure 23 shows an example of the shot boundary detection on the 30-th video of *kiss* in the TVHI dataset, which contains 5 shot boundaries in a short period. The blue bars represent the *end-frame* of each viewpoint, and the red bars represent the *start-frame* of the next viewpoint. The green axis represents the *regular-frames* within each viewpoint. The shot boundaries appear at frame 14, 50, 95, 132, and 148, accordingly.

$$\mathtt{indicator}[i] = \begin{cases} -1 & \text{if } C_1(i) = 1, C_2(i) \neq 1 \\ +1 & \text{if } C_1(i) \neq 1, C_2(i) = 1 \\ 0 & \text{if } C_1(i) \neq 1, C_2(i) \neq 1 \end{cases} \tag{4.4}$$

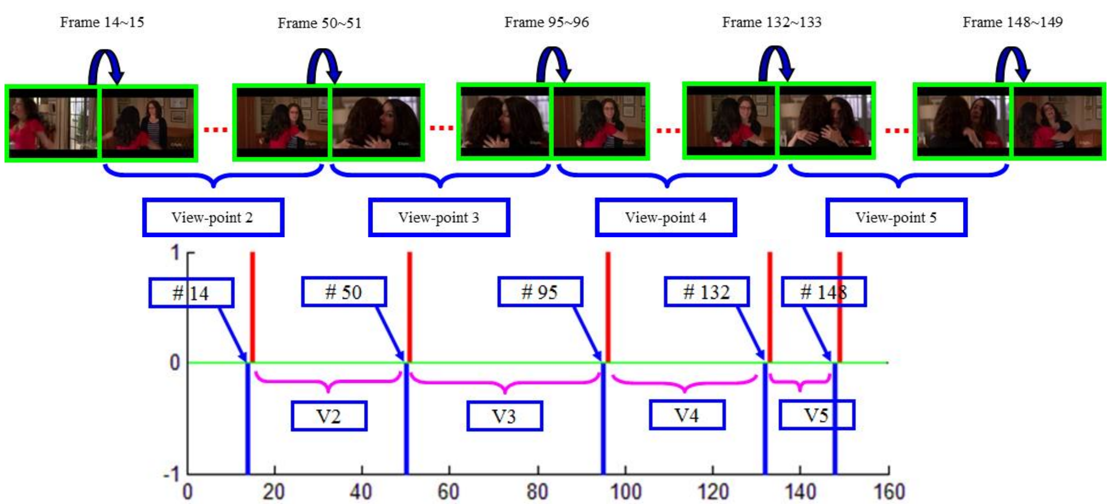


Figure 23: Examples of the shot boundary detection.

4.1.3 *Evaluation on the shot boundary detector*

We evaluate the performance of our detector on the TVHI dataset. The number of shot boundaries contained in each category are listed as follows: *handshake* (34), *highfive* (21), *hug* (60), and *kiss* (24). We use the tool [84] to compute the motion interchange pattern. The frame gap is set to 1 for generating the MIP frame by frame. For the one-class SVM, we adopt the SVDD tool provided by [85].

4.1.3.1 *Example 1: Shot boundary detection on the TVHI dataset*

In this section, we detect the shot boundaries using supervised learning. The dataset is separated into training and test parts. The training set is comprised by *handshake* and *highfive* samples, while the test set consists of *hug* and *kiss* videos. The performance of our detector can be evaluated in terms of: the number of false positive (FP) samples, the number of false negative (FN) samples, precision rate, and recall rate. For the one-class SVM, RBF kernel is selected. C and γ are the essential parameters that control the performance of the classifier. C belongs to $[1/n, 1]$, where n is the number of training samples. In this part, we set C=1 and γ=0.8.

The detection results are shown in Table 20. Although there are some frames being mis-labeled as *end-frame* or *start-frame*, the shot boundary detection is still very accurate when considering the salient pattern that consists of the *end-frame* and *start-frame* aligned in the temporal order. For completion, we switch between the training and test sets, the corresponding detection results are listed in Table 21.

Table 20: Shot boundary detection results on *hug* and *kiss* using *handshake* and *highfive* for training

	FP	FN	Precision	Recall
hug	0	1	100%	98.3%
kiss	3	1	88.4%	95.8%

Table 21: Shot boundary detection results on *handshake* and *highfive* using *hug* and *kiss* for training

	FP	FN	Precision	Recall
handshake	2	3	93.9%	91.1%
highfive	0	5	100%	85.3%

To further validate our approach, we test the performance of our detector in terms of the following configuration. Group1: *handshake* for training and the rest for testing; Group2: *handshake* and *highfive* for training, *hug* and *kiss* for testing; Group3: *handshake*, *highfive* and *hug* for training, *kiss* for testing. We compute the overall FP, FN, precision rate, and recall rate on the test sets, and the corresponding results are presented in Table 22.

4.1.3.2 *Example 2: Illumination change detection*

Apart of the shot boundary detection, we also find some special patterns that are different from the shot boundary, but not caused by the abrupt changes of viewpoint, like illumination changes. The related examples are shown in Figure 24 and 25,

Table 22: Boundary detection results through the one-class SVM

	FP	FN	Precision	Recall
Group 1	2	6	98.0%	94.3%
Group 2	3	2	96.5%	97.6%
Group 3	3	1	88.5%	95.8%

where the illumination changes are caused by the drop of the pendant lamp and the flashlight of the camera, respectively. As there are not enough sample frames of illumination changes in the TVHI dataset, we merely give some warmings when these abnormal patterns come out.

Figure 24: Examples of illumination changes caused by the sudden drop of the pendant lamp.

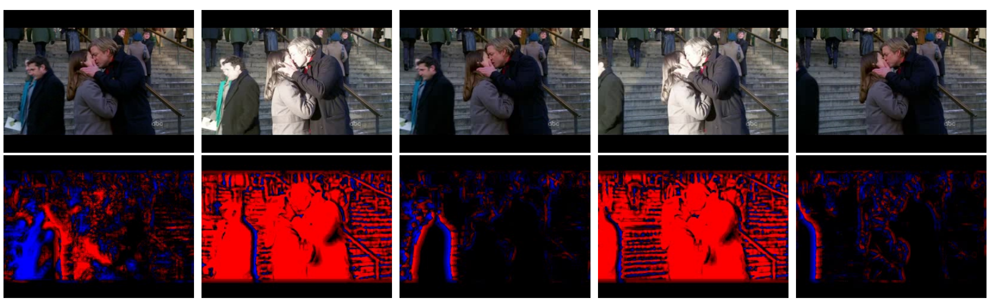


Figure 25: Examples of illumination changes caused by the flashlight of the camera.

4.2 FEATURE CONSTRUCTION

4.2.1 *Frame-based feature vector*

Although the ROI that covers the two interacting people can highlight the region where an interaction is ongoing, it is inevitable that a certain number of pixels within the ROI is irrelevant to the behavior that we are interested in. Thus, we build a *motion mask* that can capture the salient pixels corresponding to distinct motions, automatically. First, we compute the MIP indicator for each pixel, and then calculate the number of non-zero bits in the MIP indicator. If the number of non-zero bits is larger than the thresohld ε, we consider the corresponding pixel to be the salient motion point. On the contrary, when the number of non-zero bits is less than ε, it suggests that the corresponding pixel does not change significantly. We set $\varepsilon = 4$ in our implementation.

Next, we build the feature vector for each frame based on the salient motion points. As fast motion and gradual camera movements are very common phenomena in TV shows, we adopt the large-displacement optical flow (LDOF) to estimate the motion information for salient pixels. The LDOF is a particular model to estimate the dense optical flow field within successive frames that contain pixels with large displacements (e.g., fast movements of hand in *handshake* or *highfive*). In addition to the intensity and gradient constancy, the LDOF also integrates descriptor matching into the coarse-to-fine variational optical flow framework. Details about the implementation of LDOF can be found in [27].

We present examples of the *motion mask* and LDOF in Figure 26 and 27. Figure 26 demonstrates the 10-*th kiss* video in the TVHI dataset. The 1st row displays frame 118, 121, 124, 127, and 130 accordingly, from the original videos. The 2nd row is the *motion mask*, where the white pixels indicate the salient motion points. The 3rd row is the illustrations of LDOF on the salient motion points. Different colors represent diverse orientations, while the intensity of color represents the magnitude of the optical flow. Figure 27 shows the video of the 20-*th handshake* in the TVHI dataset, where frame 50, 52, 58, 59, and 65 are shown, accordingly.

The orientation and magnitude of the location x in the optical flow field is computed as in Eq. (4.5), where $u(x)$ and $v(x)$ correspond to the velocity components in horizontal and vertical directions, respectively:

$$\begin{aligned} \theta &= tan^{-1}(v(x)/u(x)) \\ mag &= \sqrt{u(x)^2 + v(x)^2} \end{aligned} \tag{4.5}$$

We build a 30-bin (as suggested in [87]) histogram of the oriented LDOF (denoted as HO-LDOF) on the basis of salient pixels, where each bin covers 12 degrees. Each salient pixel is assigned to the bin corresponding to its optical flow orientation, and weighted by the magnitude. The obtained histogram is normalized using l_2 norm, to generate the feature vector for each frame.

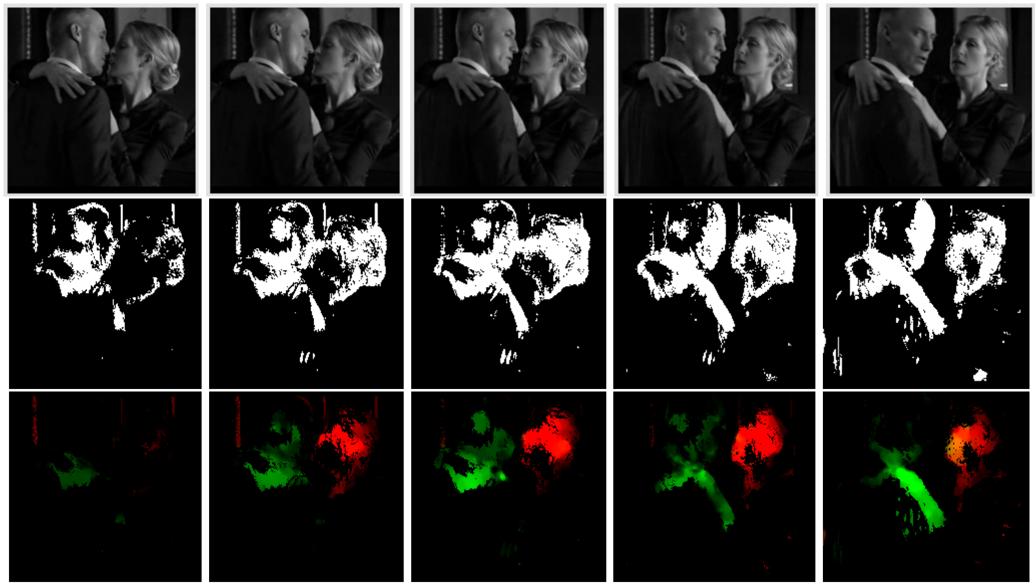


Figure 26: Examples of the *motion mask* and LDOF on *kiss*.

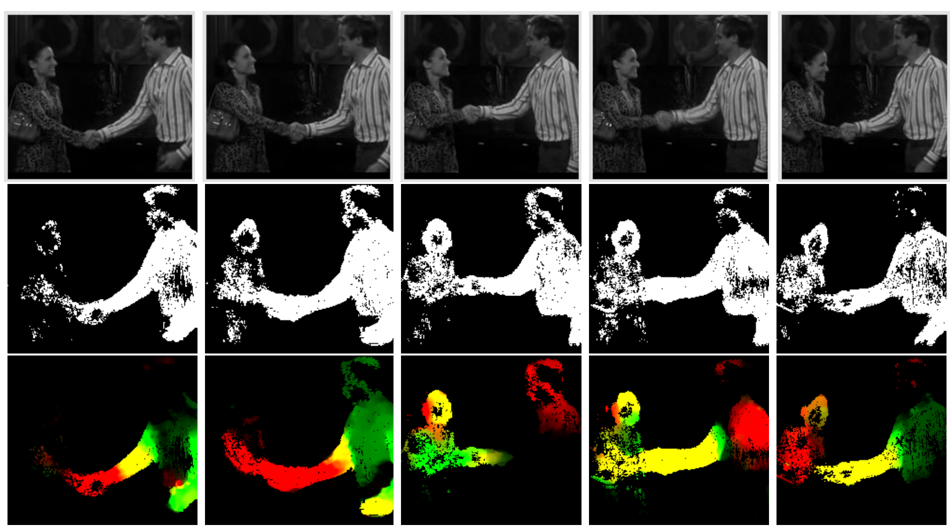


Figure 27: Examples of the *motion mask* and LDOF on *handshake*.

4.2.2 *Video-based feature vector*

We exploit the self-similarity matrix (SSM) to model the temporal correlation of human interactions. SSM descriptors have proved to be stable features under different viewpoints for the same action performing by diverse people [88].

For a sequence of images $\mathcal{I} = \{I_1, I_2, ..., I_T\}$, a SSM of $\mathcal{I}$ is a square symmetric matrix of size $T \times T$:

$$[e_{ij}]_{i,j=1,2,...,T} = \begin{bmatrix} 0 & e_{12} & e_{13} & \cdots & e_{1T} \\ e_{21} & 0 & e_{23} & \cdots & e_{2T} \\ e_{31} & e_{32} & 0 & \cdots & e_{3T} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e_{T1} & e_{T2} & e_{T3} & \cdots & 0 \end{bmatrix} \tag{4.6}$$

where e_{ij} is the distance between a certain low-level feature extracted in frame I_i and I_j, respectively. The exact structure of this matrix depends on the feature and the distance measure used for computing the entries e_{ij}. In this section, we use the Euclidean distance and frame-based feature discussed in Section 4.2.1 to construct the self-similarity matrix. An example of SSMs is shown in Figure 28.

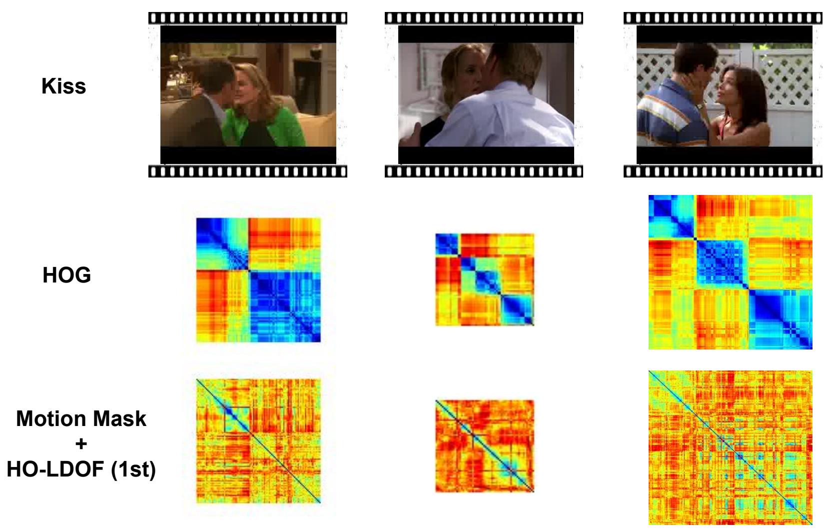


Figure 28: Example of *kiss*. SSMs obtained using the HOG information (2nd row) and the proposed method (3rd row).

Once SSMs have been computed, the same strategy described in [88] is adopted for calculating local descriptors at multiple temporal scales. For each SSM diagonal point, three local descriptors are computed corresponding to three different diameters of the log-polar domain (namely, 28, 42 and 56 frames in diameter). After extracting the SSM descriptors, classification is done through the '*bag-of-words*+SVM' strategy. Moreover, different encoding schemes (apart of the *bag-of-words* representation) can also be applied on the SSM descriptors to generate the feature vector for each video. The comparison of diverse encoding strategies will be discussed in Section 4.3.

In addition to the stabilities mentioned above, another advantage of using SSM is the suitability of modeling the temporal correlation of human activities in short video clips. Although the traditional models, such as HMM and CRF, have already been applied to human behavior understanding, several constraints still need to be satisfied. For example, the videos should cover the whole evolution of a certain activity, which implies that the model has to contain all the hidden states and the possible transitions among different states. However, this is not always verified in the TVHI dataset, where some videos only include the very essential portion of the interaction, while others also include many frames that are not related to the ongoing interaction. Moreover, how to fix the number of hidden states is still an open issue, and it is usually determined using cross-validation or defined by prior knowledge. The adoption of SSM can cope with these limitations to some degree, for instance, even when a few parts of a behavior are missing in a video, the SSM can still preserve a certain portion of distinct visual patterns.

4.3 EVALUATION

Dataset: The TVHI dataset is used to validate this approach.
Comparison methods: We adopt the standard STIP approach as the baseline. Moreover, we also compute the self-similarity matrices on the basis of other frame-based features in the region of interest: (1) HOG; (2) HOF based on the Farneback's optical flow [89] (denoted as 'Dense HOF'); (3) HO-LDOF in the whole region of interest without salient motion point detection (denoted as 'Dense HO-LDOF'). Specifically, we compare both the 1st and 2nd order optical flow when computing the HO-LDOF. The 1st order optical flow highlights the variations in a frame, whereas, the 2nd order optical flow emphasizes the motion boundaries as discussed in [37].
Setup: We randomly split the dataset into training and test sets 10 times, where each category contains 25 videos for training and the rest for testing. Negative samples that do not contain any kind of interaction are not considered in the classification task. The average classification accuracy of different frame-based features are presented in Figure 29. The last two items are the results of our approach, where we adopt the *motion mask* to select the salient motion points in the region of interest, and compute the histogram of oriented LDOF in terms of 1st order and 2nd order, respectively.
Results: From Figure 29, we can conclude: (1) By combining SSM descriptors with the *bag-of-words* representation, both HOG and dense HOF achieve better

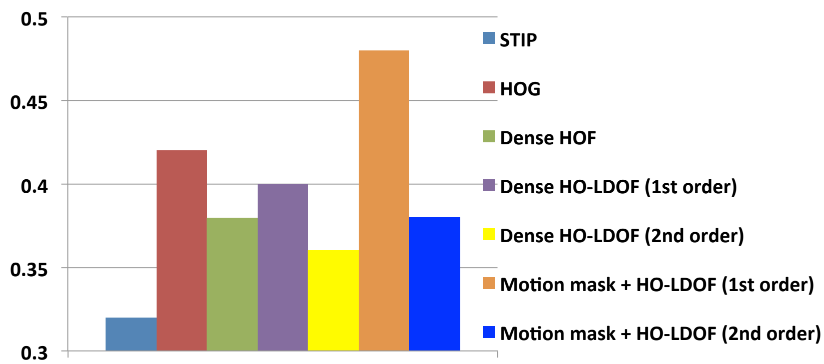


Figure 29: The average classification accuracy using different frame-based features.

results compared to the standard STIP method; (2) The performance of dense HOF is worse than HOG and dense HO-LDOF(1st order), as the estimation of optical flow is not accurate in unconstrained videos that contain fast motion and gradual camera movements; (3) The performance of 2nd order HO-LDOF is worse than 1st order HO-LDOF, but still better than the STIP feature; (4) *Motion mask* allows extracting better salient motion points, and further improves the classification performance.

Next, we compare our method with the recent work presented in [62], which exploits different STIP-based models for interaction recognition. The split of the dataset adopts the standard training/test partitions as suggested in [58]. The classification performance is measured in terms of the average precision rate (AP) as listed in Table 23.

Table 23: Comparison of different STIP-based models and our method for human interaction recognition

Method	AP
Harris3D STIP + k-means	0.3551
Dense STIP + k-means	0.3923
Dense STIP + Random Dictionary	0.3859
Dense STIP + Compressed Dictionary	0.3854
Dense STIP + Class-specific Dictionary	0.3689
Our approach	**0.4833**

Finally, we combine the SSM descriptor with different encoding strategies. The classification results are presented in Table 24, where the *Fisher* encoding scheme achieves the best performance. The corresponding confusion matrix of using the *Fisher* encoding is presented in Figure 30, where *handshake* and *hug* achieve the best classification accuracy. For *highfive*, a large portion (30%) is mis-classified as *handshake*. This is because some patterns of hand movements are shared. *Kiss* is confused with *hug* to a great extent (37%), which is due to the fact that in many videos, *hug* is only a sub-pattern of *kiss*, thus making it difficult to separate them properly.

Table 24: Comparison of different encoding strategies using the frame-based feature of 'Motion mask + LDOF (1st order)'

Encoding Method	Average accuracy
Bag-of-words (BOW)	48.1%
Kernel Codebook (KCB)	49.0%
Fisher Encoding (FK)	**50.1**%
Locality constrained linear coding (LLC)	49.6%

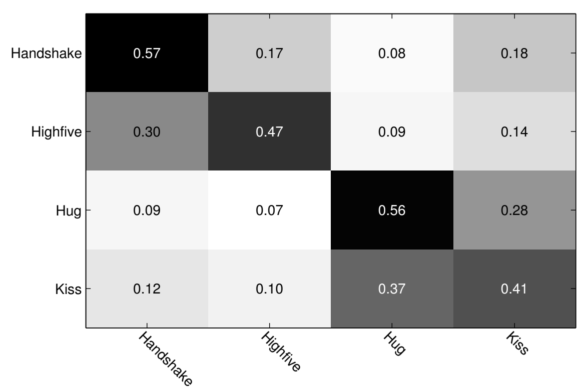


Figure 30: Confusion matrix on the TVHI dataset based on 'Motion Mask+HO-LDOF (1st order)' using the *Fisher* encoding scheme.

5

INTERACTION RECOGNITION THROUGH MULTIPLE-INSTANCE-LEARNING APPROACH

In this chapter, we propose a new framework to recognize human interactions in realistic scenarios. At the low level, we extract trajectories to represent human motion. Then, the coherent filtering algorithm [28] is exploited to cluster the trajectories into different groups, which named as *local motion patterns*. These local motion patterns don't exhibit regular shapes like cuboid, cylinder, sphere. Instead, they are more closely related to fluid or manifold, and may correspond to perceptual meaningful body movements, for instance, *raising arms*, *shaking hands*, and *stretching legs*, etc. Each local motion pattern consists of many different trajectories that have the same or similar motion trend. For simplicity, we only take the central point (can be viewed as a 'particle') along each trajectory as its representation (see Figure 31), thus the local motion pattern can be considered as an ensemble of particles. We compute the histogram of the large displacement optical flow [27] (denoted as HO-LDOF) on these particles as the group motion feature. Therefore, each video contains a variety of different local motion patterns represented by HO-LDOF.

For categorization, the multiple-instance-learning (MIL) is exploited in our work. MIL is a supervised learning framework that deals with uncertainty of instance labels, where training data is available as bags of instances with labels only for the bags. Instance labels in a bag remain unknown, and might be inferred during the learning procedure. A positive bag must contain at least one instance that labeled as 'positive', while instances in a negative bag are altogether labeled as 'negative'. In our application, we adopt the citation-KNN (C-KNN) [90], which is a typical MIL algorithm for human interaction recognition. Each video can be viewed as a 'bag', and the local motion patterns are considered to be its 'instances'. Classification is done through the one-against-one manner, where we train $n(n\text{-}1)/2$ binary classifiers using C-KNN (n indicates the total number of interaction classes). The final decision is determined by majority voting. We validate our approach on two human interaction benchmarks, namely, the TV human interaction dataset and

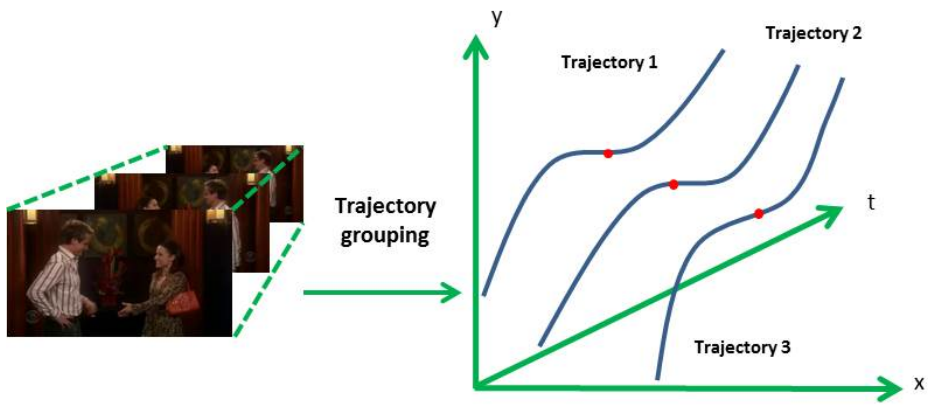


Figure 31: Trajectory representation: the blue curves represent trajectories taken from the same group, while the red dots represent the central points of trajectories along the time axis.

the UT human interaction dataset. Details are further introduced in the experimental section.

To summarize, the main contributions of this work are: (1) we adopt the coherent filtering to cluster trajectories, thus generating perceptual meaningful motion patterns, which is known as *local motion patterns*; (2) we create an efficient feature representation for the *local motion pattern*, and adopt MIL for recognition, which greatly improve the classification performance. The proposed framework is presented in Figure 32.

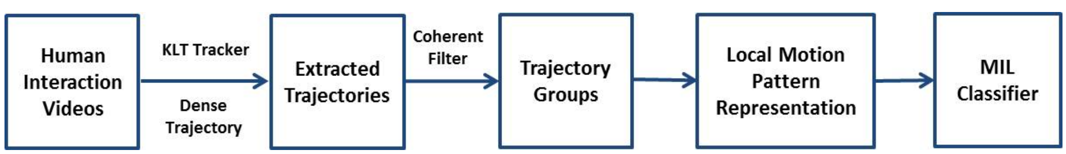


Figure 32: The proposed framework

5.1 TRAJECTORY EXTRACTION

Typical strategies for trajectory extraction include: (1) KLT tracklets, and (2) dense trajectories. In Figure 33 and 34, we visualize the differences between the above two methods when extracting trajectories from the same video. It can be seen clearly that dense trajectories allow for a better motion representation, which can reveal distinct perceptual patterns, while the KLT tracklets are more 'noisy', exhibiting irregular trajectories.

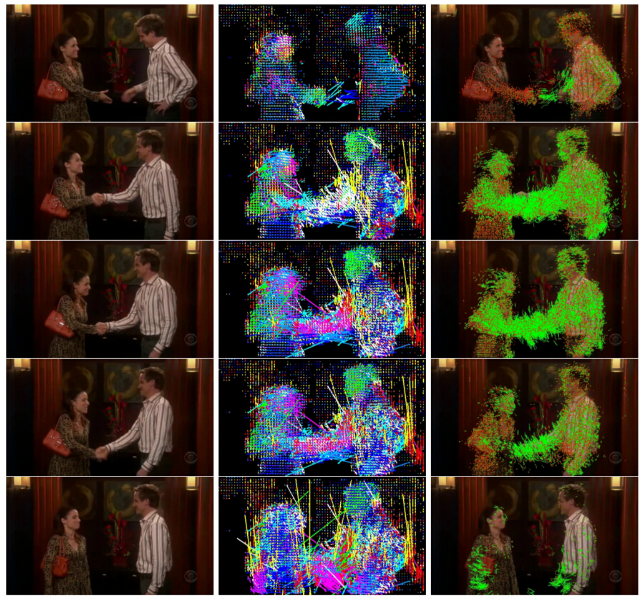


Figure 33: Examples of trajectory extraction: *handshake* video.

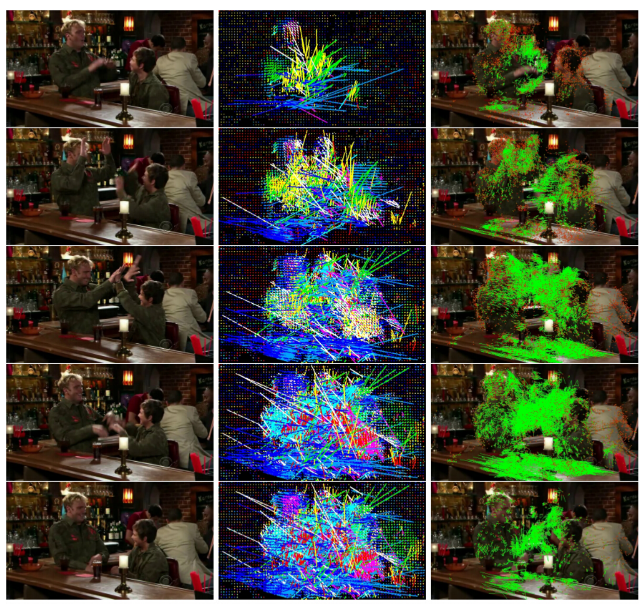


Figure 34: Examples of trajectory extraction: *highfive* video.

5.2 LOCAL MOTION PATTERNS

The total number of trajectories extracted in a video varies depending on its length and contents, usually ranging from several hundreds to several thousands on the standard activity benchmarks.

The traditional way of using trajectories for activity recognition follows the ’motion descriptors + *bag-of-words*’ scheme. For example, in the dense trajectory based method, the motion boundary histogram (MBH) is first computed for each trajectory. Then, the *k-means* clustering is applied on all the MBH descriptors in the training set, thus generating a so-called visual codebook comprised by the cluster centers. Next, each video is represented using a normalized feature vector by quantizing its MBH descriptors onto the codebook. Finally, the standard SVM is adopted for classification.

However, there are some weak points in the above strategy: (1) the *k-means* clustering is a very coarse representation of signals, thus losing a lot of essential information when constructing the visual codebook; (2) for a given video, quantizing is a global operation, which can not preserve the local motion characteristics, as for instance, the temporal order of body movements; (3) not all the trajectory descriptors are useful in the classification task. Some irrelevant trajectories can be viewed as ’outliers’ in building the visual codebook.

Due to the limitations mentioned above, we propose a new strategy to recognize human interactions. First, we adopt the coherent filtering (CF) algorithm to generate the trajectory clusters, in which trajectories with the joint or similar motion trend are grouped. This operation helps getting rid of the side-effects of non-relevant trajectories.

The coherent filtering is first proposed in [28] with the aim of coherent motion detection in noisy time-series data, such as crowd analysis in public spaces. The local spatio-temporal relationship of individuals (particles) in coherent motion follows two key properties, known as *Coherent Neighbor Invariance*: (1) invariance of spatio-temporal relationships, and (2) invariance of velocity correlations. Based on the *Coherent Neighbor Invariance*, the coherent filtering finds the invariant neighbors and pairwise connections of particles, thus generating the coherent motion clusters. Finally, these clusters are associated and updated over successive frames.

Examples of trajectory grouping using the coherent filtering are shown in Figure 35, where videos from the TVHI dataset are used for display (from top to bottom: *handshake*, *highfive*, *hug*, and *kiss*). The colored particles correspond to trajectory centers shown in Figure 31. Particles with the same color fall into the same group, and exhibit a similar motion trend.

We named the trajectory clusters as *local motion patterns*. Usually, local motion patterns may correspond to some perceptual meaningful body movements, such

Figure 35: Examples of trajectory grouping using the coherent filtering.

as *stretching arms* or *shaking hands*. We demonstrate the dynamic evolution of trajectory groups of human interactions in Figure 36 and 37, respectively.

We then use the histogram of large-displacement optical flow (HO-LDOF) as the feature vector for each local motion pattern, in order to deal with the problem of fast body movements in realistic scenarios, thus each video can be expressed by a collection of local motion patterns that described by HO-LDOF.

5.3 CITATION-KNN FOR INTERACTION RECOGNITION

Considering the limitation of the *k-means* clustering, we adopt the multiple-instance-learning strategy for classification. In this case, local motion patterns are no longer

Figure 36: Example 1. Evolution of trajectory group, *handshake* video.

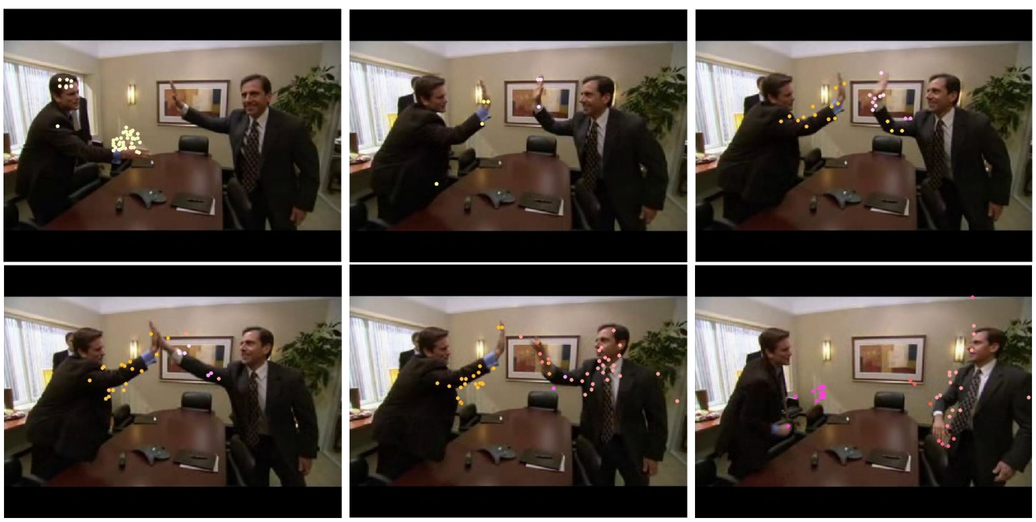


Figure 37: Example 2. Evolution of trajectory group, *highfive* video.

used to generate the visual codebook. Alternatively, we consider each video as a 'bag', in which each local motion pattern is treated as a 'instance'.

After generating the local motion patterns in videos, we adopt the so-called 'citation-KNN' (C-KNN) algorithm for classification. C-KNN is a specific nearest neighborhood algorithm that deals with multiple instance learning problems. It predicts the label of an unlabeled bag based on the nearest neighborhood approach.

Instead of using the traditional Euclidean distance, the algorithm defines a new bag-level metric that measures the distance between two different bags using the *minimum Hausdorff distance*, which is denoted as $\mathtt{Dist}(A, B)$ in (5.1):

$$\mathtt{Dist}(A, B) = \min_{\forall a_i \in A, \forall b_j \in B} (\mathtt{dist}(a_i, b_j)) \tag{5.1}$$

where A and B represent two different bags, a_i and b_j are the instances from their corresponding bags, and $\mathtt{dist}(a_i, b_j)$ measures the Euclidean distance between a_i and b_j. It can be seen from (5.1) that $\mathtt{Dist}(A, B)$ is the shortest distance between any pair of instances taken from their respective bags.

After defining the distance between bags, the citation approach is adopted for classification. For a given bag M, the algorithm considers not only the bags in its neighborhood (known as 'references'), but also the bags that view M as their neighbors (known as 'citers'). After calculating the summation of R-nearest references and C-nearest citers in terms of positive bags (denoted as 'P') and negative bags (denoted as 'N'), respectively, the label of bag M can be determined as: *positive* (if $P>N$), *negative* otherwise.

We exploit the one-against-one strategy to deal with the multi-classes problem, where we collect the results from all the binary classifiers and make the final decision using majority voting.

5.4 EVALUATION

In this section, we validate our approach on two standard human interaction benchmarks, namely: the TV human interaction dataset and the UT human interaction dataset. Considering the limited number of sample videos in these two datasets, we use the leave-one-out cross-validation strategy to evaluate the classification performance. For C-KNN, we adopt the implementation proposed in [91] to train the binary classifiers.

5.4.1 *TV Human Interaction Dataset*

The TVHI dataset consists of four different types of human interactions, namely *handshake*, *highfive*, *hug*, and *kiss*, where each class contains 50 video clips. We first compare our approach with the standard STIP-based and dense trajectory based approaches, respectively. For STIP-based method, spatio-temporal interest points are captured using the detector proposed in [30], where each interest point is described using the concatenation of histogram of gradient (HOG) and histogram of oriented optical flow (HOF), with the dimensionality of the feature vector equal to 72+90=162 in total. For dense trajectory based method, we adopt the MBH descriptor as the motion feature for each trajectory, with the dimensionality equal to $96 \times 2 = 192$. Then, the standard *bag-of-words* scheme is used for classification. We compare our approach with several baseline results in Table 25.

Table 25: Comparison with STIP and dense trajectory based methods

Motion Representation	**Descriptor**	**Codebook**	**Accuracy**
STIP	HOG+HOF	size=500	45.5 %
Dense Trajectory	MBH	size=50	53.0 %
Dense Trajectory	MBH	size=100	56.5 %
Dense Trajectory	MBH	size=500	53.5 %
Local Motion Pattern (KLT tracklets)	HO-LDOF	N/A	**68.5 %**
Local Motion Pattern (dense trajectories)	HO-LDOF	N/A	**71.5 %**

Secondly, we compare the multiple-instance-learning strategy with the *bag-of-words* scheme. Here, we use dense trajectories to generate the *local motion patterns*. Then, the HO-LDOF is quantized into 18 bins, in which each bin covers 20 degrees. We apply the *k-means* clustering on the *local motion pattern* descriptors (namely, HO-LDOF) to construct the codebook, and use the standard SVM for classification as the baseline. For comparison, we regard each *local motion pattern* as the 'instance' in a video, and then adopt the C-KNN for classification. The results are shown in Table 26, from where we can conclude that the MIL algorithm outperforms the *bag-of-words* scheme significantly.

Table 26: Comparison using different classification strategies

Trajectory Type	Dense trajectory	Dense trajectory
Local Motion Pattern Descriptor	HO-LDOF	HO-LDOF
Classification Scheme	BoW+SVM	C-KNN
Average Accuracy	49.5%	**71.5 %**

Finally, we compare the classification performance using different low level motion representation, namely, KLT tracklets and dense trajectories, respectively. Specifically, we set a threshold T to control the selection of trajectory groups. In case a group contains a limited number of trajectories (less than T), it will not be considered in the classification. The final results are reported in Table 27 and 28, respectively. From these two tables we can observe that: (1) on the TVHI dataset, dense trajectories are slightly better in the classification as compared to the KLT tracklets, no matter how the threshold T is set; (2) increasing T, the classification performance decreases, which implies that even groups containing only a few trajectories contribute to the classification task. Additionally, it is worth pointing out that although dense trajectories can achieve better classification performance, KLT tracklets run much faster when extracting trajectories, requiring to find a trade-off between speed and accuracy in realistic applications.

Table 27: Average accuracy using our approach on the TVHI dataset (KLT tracklets)

	HandShake	HighFive	Hug	Kiss	Accuracy
KLT, T=20	66%	66%	60%	66%	64.5%
KLT, T=1	74%	72%	60%	66%	68.0%

Table 28: Average accuracy using our approach on the TVHI dataset (dense trajectories)

	HandShake	HighFive	Hug	Kiss	Accuracy
Dense, T=20	66%	72%	60%	62%	65.0%
Dense, T=1	78%	70%	58%	80%	71.5%

5.4.2 *UT Human Interaction Dataset*

The UT dataset is designed for recognizing high-level two-person interactions from surveillance cameras. The videos in this dataset are further divided into two subsets, where videos in SET 1 are taken on a parking area with almost static background and little camera jitters, while videos in SET 2 are taken on a lawn in a windy day with moving background and more camera jitters. In SET 1, only a pair of people interacts, while SET 2 is more challenging due to multiple people moving in the scenes and background clutters. This dataset consists of 5 different types of human interactions, namely, *handshake*, *punch*, *push*, *kick*, and *hug*, each of which contains 10 sample videos in each subset. Moreover, the videos are recorded under a constrained condition with fixed camera viewpoint, moderate people scale, and staged interactions. We validate our approach on this dataset using dense trajectories and set the threshold T=1 (which has already proved to be a better strategy on the TVHI dataset). We compare our approach with the STIP-based approach and the structural learning methods proposed in [58]. The results are presented in Table 29 and 30, respectively.

From Table 29 and 30 we can find: (1) on this constrained dataset, our method is comparable with the STIP-based approach. We are slightly worse on SET 1 (76% VS 82%), but much better on SET 2 (78% VS 66%), which suggests that our method is

Table 29: Average accuracy on SET 1. $(*)$ indicates that the low level features (i.e., upper bodies, head poses) in the corresponding approach are annotated manually.

	Punch	Hug	Kick	Push	Handshake	Accuracy
STIP-based approach	80 %	90%	90%	80%	80%	82.0%
Structural learning using local context $^{(*)}$	60 %	50%	50%	100%	90%	70.0%
Structural learning using full strcuture $^{(*)}$	60 %	100%	80%	80%	100%	84.0%
Our approach	90 %	50%	80%	80%	80%	76.0%

Table 30: Average accuracy on SET 2. $(*)$ indicates that the low level features (i.e., upper bodies, head poses) in the corresponding approach are annotated manually.

	Punch	Hug	Kick	Push	Handshake	Accuracy
STIP-based approach	70 %	80%	60%	40%	80%	66.0%
Structural learning using local context $^{(*)}$	10 %	100%	50%	90%	80%	66.0%
Structural learning using full structure $^{(*)}$	70 %	90%	90%	90%	90%	86.0%
Our approach	100 %	50%	90%	70%	80%	78.0%

more suitable for the realistic scenarios; (2) compared to the structural learning based approaches, we are better than the structural learning that incorporates local context information (in [58], the ’local context information’ is represented by computing the HOG descriptors in the regions of interest that surrounds upper bodies). Structural learning using full structure information (in [58], the ’full structure’ refers to head poses, local context information, and global context information, such as ’far’, ’near’, ’adjacent’, and ’overlap’) always performs better than our approach, exceeding our results around 8% in both subsets. However, the full structure information is not always that straightforward to obtain in very challenging scenarios, which often involves complicated trackers, people detectors, and accurate object segmentation. This severely constrains the applicability of their method. In contrast, our approach does not rely on any body detector, pose estimator, etc., thus it can be applied in more realistic environments.

6

CONCLUSION

In this work, two open issues in human interaction analysis are studied, namely: (1) discriminative patch segmentation, and (2) two-person interaction classification.

For discriminative patch segmentation, two different approaches are proposed with respect to different application scenarios: (1) videos from surveillance cameras, and (2) videos from TV shows. The patches extracted by our approaches demonstrate better discriminating capability as compared to the original videos, while preserving the perceptual meaningful portion of human activities as well.

For interaction recognition, we adopt two different approaches. The first one exploits the motion interchange pattern to capture salient motion points in a video, and then uses the SSM descriptors for classification. This method can analyze human interactions in the wild, which allows for camera motion and changes of viewpoint in unconstrained environments. The adoption of the self-similarity matrix can handle the temporal correlation modeling in short duration videos that do not capture sufficient contextual information. The second solution adopts trajectories as motion features. After clustering the trajectories into perceptual meaningful groups, classification is achieved using the C-KNN algorithm, which is a typical multiple-instance-learning approach. The good classification performance is obtained due to: (1) the efficient motion representation, namely the histogram of oriented LDOF, and (2) the adequate selection of classifiers.

For the future work, we will first apply the discriminative patch segmentation algorithms to the area of video summarization. Then, for interaction recognition, we will focus on modeling the interaction dynamics for two-person interaction recognition. Currently, we adopt a novel joint Schatten p-norm model, which considers human interaction as an interactive behavior that comprised by two independent individuals communicating with each other. This model can learn the joint interactive patterns between the two interacting persons and the distinctive interactive patterns with respective to each individual, simultaneously. The preliminary results of this method have shown better performance as compared to the approach proposed in Chapter 4. Regarding the MIL, we prepare to apply the current approach to large-scale datasets (i.e., Hollywood2 dataset, HMDB). The main issue we need to consider is to reduce the number of trajectory groups in long video clips, as the complexity of the C-KNN algorithm is high. One possible solution is applying the unsupervised clustering algorithms to the trajectory groups (represented by HO-LDOF) in a video, and using the clustering centers as the representative *local motion patterns*. In this way, each video can be described using a limited number of typical *local motion patterns*, thus speeding up the classification procedure.

7

PUBLICATIONS

[1] **Bo Zhang**, Nicola Conci, and Francesco G. B. De Natale. Segmentation of discriminative patches in human activity videos. ACM Transactions on Multimedia Computing, Communications and Applications (ACM TOMM, and formerly known as TOMCCAP, 2015).

[2] **Bo Zhang**, Paolo Rota, Nicola Conci, and Francesco G. B. De Natale. Human interaction recognition in the wild: analyzing trajectory clustering from multiple-instance-perspective. 2015 IEEE International Conference on Multimedia and Expo (IEEE ICME).

[3] **Bo Zhang**, Yan Yan, Nicola Conci, and Nicu Sebe. You talkin' to me? Recognizing complex human interactions in unconstrained videos. The 22nd ACM International Conference on Multimedia (ACM MM), Orlando, 3-7 November 2014. DOI: http://dx.doi.org/10.1145/2647868.2654996.

[4] **Bo Zhang**, Nicola Conci, and Francesco G. B. De Natale. Camera viewpoint change detection for interaction analysis in TV shows. The 21st IEEE International Conference on Image Processing (IEEE ICIP), Paris, 27-30 October 2014. DOI:10.1109/ICIP.2014.7025515.

[5] **Bo Zhang**, Nicola Conci, and Francesco G. B. De Natale. Human interaction recognition through two-phase sparse coding. SPIE Conference on Video Surveillance and Transportation Imaging Applications, 2014. DOI:10.1117/12.2041206.

[6] **Bo Zhang**, Francesco G. B. De Natale, and Nicola Conci. Recognition of social interactions based on feature selection from visual codebooks. The 20th IEEE International Conference on Image Processing (IEEE ICIP), Melbourne, 15-18 September 2013. DOI:10.1109/ICIP.2013.6738734.

[7] **Bo Zhang**, Paolo Rota, and Nicola Conci. Recognition of two-person interaction in multi-view surveillance video via proxemics cues and spatio-temporal interest points. SPIE Conference on Video Surveillance and Transportation Imaging Applications, 2013. DOI: 10.1117/12.2003686.

ACKNOWLEDGEMENT

BIBLIOGRAPHY

[1] W.Y. Lin, M.T. Sun, R. Poovandran, and Z.Y. Zhang. Human activity recognition for video surveillance. In *Proceedings of International Symposium on Circuits and Systems*, pages 2737–2740. IEEE, 2008.

[2] M. Cristani, R. Raghavendra, A. Del Bue, and V. Murino. Human behavior analysis in video surveillance: A social signal processing perspective. *Neurocomputing*, 100:86–97, 2013, Elsevier.

[3] M. Hoai and F. De la Torre. Max-margin early event detectors. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 2863–2870. IEEE, 2012.

[4] Y. Ke, R. Sukthankar, and M. Hebert. Volumetric features for video event detection. *International Journal of Computer Vision*, 88(3):339–362, 2010, Springer.

[5] Y. Cong, J. S. Yuan, and J. Liu. Sparse reconstruction cost for abnormal event detection. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 3449–3456. IEEE, 2011.

[6] M. Rodriguez. Cram: Compact representation of actions in movies. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 3328–3335. IEEE, 2010.

[7] Y. J. Lee, J. Ghosh, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 1346–1353. IEEE, 2012.

[8] Y. G. Jiang, Z. G. Li, and S. F. Chang. Modeling scene and object contexts for human action retrieval with few examples. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(5):674–681, 2011.

[9] J. Simon and L. Shao. Content-based retrieval of human actions from realistic video databases. *Information Sciences*, 236:56–65, 2013, Elsevier.

[10] J. K. Aggarwal and M. S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys (CSUR)*, 43(3):16, 2011, ACM.

[11] S. R. Fanello, I. Gori, G. Metta, and F. Odone. Keep it simple and sparse: real-time action recognition. *The Journal of Machine Learning Research*, 14(1):2617–2640, 2013, MIT Press.

[12] C. Y. Wang, Y. Z. Wang, and A. L. Yuille. An approach to pose-based action recognition. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 915–922. IEEE, 2012.

[13] A. Yao, J. Gall, and L. Van Gool. Coupled action recognition and pose estimation from multiple views. *International Journal of Computer Vision*, 100(1):16–37, 2012, Springer.

[14] H. Z. Ning, W. Xu, Y. H. Gong, and T. Huang. Latent pose estimator for continuous action recognition. In *Proceedings of European Conference on Computer Vision*, pages 419–433. Springer, 2008.

[15] G. Yu, J. S. Yuan, and Z. C. Liu. Propagative hough voting for human activity recognition. In *Proceedings of European Conference on Computer Vision*, pages 693–706. Springer, 2012.

[16] M. S. Ryoo and J. K. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *Proceedings of International Conference on Computer Vision*, pages 1593–1600. IEEE, 2009.

[17] Y. Kong, Y. D. Jia, and Y. Fu. Learning human interaction by interactive phrases. In *Proceedings of European Conference on Computer Vision*, pages 300–313. Springer, 2012.

[18] Y. Kong, Y. D. Jia, and Y. Fu. Interactive phrases: Semantic descriptions for human interaction recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 36, pages 1775–1788. 2014.

[19] M. J. Marín-Jiménez and N. P. de la Blanca. Human interaction recognition by motion decoupling. In *Pattern Recognition and Image Analysis*, pages 374–381. 2013, Springer.

[20] B. B. Zhan, D. N. Monekosso, P. Remagnino, S. A. Velastin, and L. Q. Xu. Crowd analysis: a survey. *Machine Vision and Applications*, 19(5-6):345–357, 2008, Springer.

[21] B. Solmaz, B. E. Moore, and M. Shah. Identifying behaviors in crowd scenes using stability analysis for dynamical systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(10):2064–2070, 2012.

[22] R. Mehran, A. Oyama, and M. Shah. Abnormal crowd behavior detection using social force model. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 935–942. IEEE, 2009.

[23] W. G. Choi, K. Shahid, and S. Savarese. Learning context for collective activity recognition. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 3273–3280. IEEE, 2011.

[24] T. Lan, Y. Wang, W. L. Yang, S. N. Robinovitch, and G. Mori. Discriminative latent models for recognizing contextual group activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1549–1562, 2012.

[25] T. K. Huang, R. C. Weng, and C. J. Lin. Generalized bradley-terry models and multi-class probability estimates. *The Journal of Machine Learning Research*, 7:85–115, 2006, MIT Press.

[26] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf. Motion interchange patterns for action recognition in unconstrained videos. In *Proceedings of European Conference on Computer Vision*, pages 256–269. Springer, 2012.

[27] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

[28] B. L. Zhou, X. O. Tang, and X. G. Wang. Coherent filtering: detecting coherent motions from crowd clutters. In *Proceedings of European Conference on Computer Vision*, pages 857–871. Springer, 2012.

[29] P. Borges, N. Conci, and A. Cavallaro. Video-based human behavior understanding: a survey. *IEEE Transactions on Circuits System for Video Technology*, 23(11):1993–2008, 2013.

[30] I. Laptev. On space-time interest points. *International Journal of Computer Vision*, 64(2):107–123, 2005, Springer.

[31] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *2nd Joint International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72. IEEE, 2005.

[32] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *Proceedings of European Conference on Computer Vision*, pages 650–663. Springer, Marseille, France, 2008.

[33] A. Klaser and M. Marszalek. A spatio-temporal descriptor based on 3d-gradients. In *Proceedings of British Machine Vision Conference*, pages 995–1004, 2008.

[34] M. Y. Chen and A. Hauptmann. Mosift: Recognizing human actions in surveillance videos. cmu-cs-09-161, carnegie mellon university. 2009.

[35] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *Proceedings of British Machine Vision Conference*, pages 124.1–124.11, 2009.

[36] M. Raptis and S. Soatto. Tracklet descriptors for action modeling and video analysis. In *Proceedings of European Conference on Computer Vision*, pages 577–590. Springer, Heraklion, Crete, Greece, 2010.

[37] H. Wang, A. Klaser, C. Schmid, and C. L. Liu. Action recognition by dense trajectories. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 3169–3176, Colorado Springs, CO, USA, 2011. IEEE.

[38] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proceedings of International Conference on Computer Vision*, pages 3551–3558, Sydney, Australia, 2013. IEEE.

[39] A. Tamrakar, S. Ali, Q. Yu, J. G. Liu, O. Javed, A. Divakaran, H. Cheng, and H. Sawhney. Evaluation of low-level features and their combinations for complex event detection in open source videos. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 3681–3688. IEEE, 2012.

[40] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a svm approach. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 3, pages 32–36, Cambridge, UK, 2004. IEEE.

[41] M. Elad and M. Aharon. The k-svd: An algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[42] T. Guha and R. K. Ward. Learning sparse representations for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1576–1588, 2012.

[43] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[44] D. Oneata, J. Verbeek, and C. Schmid. Action and event recognition with fisher vectors on a compact feature set. In *Proceedings of International Conference on Computer Vision*, pages 1817–1824. IEEE, 2013.

[45] Douglas L Vail, Manuela M Veloso, and John D Lafferty. Conditional random fields for activity recognition. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 235:1–235:8. ACM, 2007.

[46] K. Tang, F. F. Li, and K. Daphne. Learning latent temporal structure for complex event detection. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 1250–1257. IEEE, 2012.

[47] X. D. Liang, L. Lin, and L. L. Cao. Learning latent spatio-temporal compositional model for human action recognition. In *Proceedings of the 21st ACM International Conference on Multimedia*, pages 263–272. ACM, 2013.

[48] S. Satkin and M. Hebert. Modeling the temporal extent of actions. In *Proceedings of European Conference on Computer Vision*, pages 536–548. Springer, 2010.

[49] M. Raptis and L. Sigal. Poselet key-framing: A model for human activity recognition. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 2650–2657, Portland, Oregon, USA, 2013. IEEE.

[50] L. M. Wang, Y. Qiao, and X. O. Tang. Motionlets: Mid-level 3d parts for human motion recognition. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 2674–2681, Portland, Oregon, USA, 2013. IEEE.

[51] A. Jain, A. Gupta, M. Rodriguez, and L. S. Davis. Representing videos using mid-level discriminative patches. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 2571–2578. IEEE, 2013.

[52] M. Sapienza, F. Cuzzolin, and P. Torr. Learning discriminative space-time actions from weakly labelled videos. In *Proceedings of British Machine Vision Conference*, pages 1–12, 2012.

[53] M. Sapienza, F. Cuzzolin, and P. Torr. Learning discriminative space-time actions from weakly labelled videos. *International Journal of Computer Vision*, pages 1–18, 2013, Springer.

[54] J. Zhu, B. Y. Wang, X. K. Yang, W. J Zhang, and Z. W Tu. Action recognition with actons. In *Proceedings of International Conference on Computer Vision*, pages 3559–3566. IEEE, 2013.

[55] P. Rota, N. Conci, and N. Sebe. Real time detection of social interactions in surveillance video. In *Proceedings of European Conference on Computer Vision Workshop*, pages 111–120. Springer, 2012.

[56] CMU. Cmu graphics lab motion capture database [online]. http://mocap.cs.cmu.edu/, 2013.

[57] M. S. Ryoo and J. K. Aggarwal. UT-Interaction Dataset, ICPR contest on Semantic Description of Human Activities (SDHA). http://cvrc.ece.utexas.edu/SDHA2010/Human_Interaction.html.

[58] A. Patron-Perez, M. Marszalek, I. Reid, and A. Zisserman. Structured learning of human interactions in tv shows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2441–2453, 2012.

[59] S. H. Park and J. K. Aggarwal. A hierarchical bayesian network for event recognition of human actions and interactions. *Multimedia systems*, 10(2):164–179, 2004, Springer.

[60] S. H. Park and J. K. Aggarwal. Semantic-level understanding of human actions and interactions using event hierarchy. In *Proceedings of International Conference on Computer Vision and Pattern Recognition Workshop*, page 12. IEEE, 2004.

[61] M. S. Ryoo and J. K. Aggarwal. Recognition of composite human activities through context-free grammar based representation. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1709–1718. IEEE, 2006.

[62] M.J. Marín-Jiménez, E. Yeguas, and N. P. De La Blanca. Exploring stip-based models for recognizing human interactions in tv videos. *Pattern Recognition Letters*, 34(15):1819–1828, 2013, Elsevier.

[63] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2247–2253, 2007.

[64] M. Marcin, I. Laptev, and C. Schmid. Actions in context. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 2929–2936. IEEE, 2009.

[65] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proceedings of International Conference on Computer Vision*, pages 2556–2563. IEEE, 2011.

[66] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero. A survey of video datasets for human action and activity recognition. *Computer Vision and Image Understanding*, 117(6):633–659, 2013.

[67] E. T. Hall. A system for the notation of proxemic behavior. *American anthropologist,*, 65(5):1003–1026, 1963.

[68] J. C. Yang, K. Yu, Y. H. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 1794–1801. IEEE, 2009.

[69] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[70] M. Elad and M. Aharon. Image denoising via learned dictionaries and sparse representation. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, volume 1, pages 895–900. IEEE, 2006.

[71] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[72] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, 11:19–60, 2010.

[73] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of International Conference on Machine Learning*, pages 689–696, Montreal, QC, Canada, 2009. ACM.

[74] X. X. Wang, L. M. Wang, and Y. Qiao. A comparative study of encoding, pooling and normalization methods for action recognition. In *Proceedings of Asian Conference on Computer Vision*, pages 572–585, 2012.

[75] A. Castrodad and G. Sapiro. Sparse modeling of human actions from motion imagery. *International Journal of Computer Vision*, 100(1):1–15, 2012, Springer.

[76] N. García-Pedrajas and D. Ortiz-Boyer. An empirical study of binary classifier fusion methods for multiclass classification. *Information Fusion*, 12(2):111–130, 2011, Elsevier.

[77] Julien Mairal. Sparse modeling software. INRIA, 2012. http://spams-devel.gforge.inria.fr/index.html.

[78] B. Zhang, F.G.B. De Natale, and N. Conci. Recognition of social interactions based on feature selection from visual codebooks. In *Proceedings of International Conference on Image Processing*, pages 3557–3561, Melbourne, Australia, 2013. IEEE.

[79] J. Komorowski, Z. Pawlak, L. Polkowski, and A. Skowron. Rough sets: A tutorial. *Rough fuzzy hybridization: A new trend in decision-making*, pages 3–98, 1999.

[80] J. O. Pedersen and Y. Yang. A comparative study on feature selection in text categorization. In *Proceedings of International Conference on Machine Learning*, pages 412–420, 1997.

[81] J. C. Niebles, C. W. Chen, and F. F. Li. Modeling temporal structure of decomposable motion segments for activity classification. In *Proceedings of European Conference on Computer Vision*, pages 392–405, Heraklion, Crete, Greece, 2010. Springer.

[82] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt. Support vector method for novelty detection. In *Proceedings of Annual Conference on Neural Information Processing Systems (NIPS)*, volume 12, pages 582–588. MIT Press, 1999.

[83] D. Tax and T. David. Support vector data description. *Machine learning*, 54(1):45–66, 2004, Springer.

[84] Kliper-Gross, O. and Gurovich, Y. and Hassner, T. and Wolf, L. Motion interchange patterns for action recognition in unconstrained videos. http://www.openu.ac.il/home/hassner/projects/MIP/.

[85] Chih-Chung Chang and Chih-Jen Lin. LIBSVM – A Library for Support Vector Machines. http://www.csie.ntu.edu.tw/ cjlin/libsvm/.

[86] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proceedings of European Conference on Computer Vision*, pages 25–36. Springer, 2004.

[87] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pages 1932–1939. IEEE, 2009.

[88] I. N. Junejo, E. Dexter, I. Laptev, and P. Perez. View-independent action recognition from temporal self-similarities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):172–185, 2011.

[89] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image Analysis*, pages 363–370. Springer, 2003.

[90] J. Wang and J. D. Zucker. Solving multiple-instance problem: A lazy learning approach. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1119–1125. ACM, 2000.

[91] Z. H. Zhou and M. L Zhang. Ensembles of multi-instance learners. In *Proceedings of European Conference on Machine Learning*, pages 492–502. Springer, 2003.