Projet ANR- 11-IS02-001

MEX-CULTURE / Multimedia libraries indexing for the preservation and dissemination of the Mexican Culture

Deliverable ED3.3

Final report on summarization and scalable search

Programme Blanc International II- 2011 Edition

IDENTIFICATION

Project acronym	MEX-CULTURE
Project title	Multimedia libraries indexing for the preservation and dissemination of the Mexican Culture
Coordinator of the French part of the project (company/organization)	Centre d'Etude et de Recherche en Informatique et Communications – Conservatoire National des Arts et Métiers
Coordinator of the Mexican part of the project (company/organization)	Instituto Politécnico Nacional - Centro de Investigación y Desarrollo de Tecnología Digital
Project coordinator (if applicable)	Michel Crucianu
Project start date Project end date	01/01/2012* 30/11/2015
Competitiveness cluster labels and contacts (cluster, name and e-mail of contact)	Cap Digital Paris-Région Philippe Roy Philippe.Roy@capdigital.com
Project website if applicable	http://mexculture.cnam.fr

* *The Mexican partners were only financed since November 2012.*

Coordinator of this report	
Title, first name, surname	Michel CRUCIANU
Telephone	+33 1 40 27 24 58
E-mail	Michel.Crucianu@cnam.fr
Date of writing	15/12/2015

Rédacteurs :	Michel Crucianu(CEDRIC-Cnam),Andrei Stoian (CEDRIC-Cnam) Gabriel Sargent (LABRI, CEDRIC-Cnam) Jenny Benois-Pineau(LABRI), Henri Nicolas(LABRI), Karina Perez-Daniel (LABRI), Sofian Maabout(LABRI) Mireya García-Vázquez (CITEDI-IPN), Alejandro Ramírez-Acosta (CITEDI-IPN)

Contents

1 Scalable Summarization

1.1 Introduction

Video documentaries are one way to capture the cultural heritage of a country and they can be used for the preservation and dissemination of the culture. However, large volumes of such content as well as their large duration enhance the necessity of developing a fast, easy and multidimensional access to them. Video summarization is a compact representation of video content, which provides access to most relevant information based on similarities.

Due to the importance of the problem of browsing and retrieving information in large data, several video summarization approaches have been proposed [1, 5, 27] and a specific task of TRECVID campaign such as rushes summarization was run. Despite the fact that numerous solutions have been proposed for this task, it still remains open, as very much application dependent and the most recent works [21] witness of the importance of this problem in multimedia research. Summarization is usually done by grouping similar video segments on the basis of continuous audio channel [33]. Generally speaking video summarization approach can be inspired by data analysis techniques such as clustering or supervised learning, *etc.*

It is crucial to incorporate the video summarization approaches into large scale multimedia applications. However, the strong requirements of those applications in terms of scale, time response and high dimensional information make the scalability a very challenging problem.

The scalability can be seen as the ability of proposed approach to generalize on a large scale of data. Another interpretation comes from multi-view data representation and means that the data can be described in a coarse-to-fine manner, this is how we understand the scalable video summarization. A scalable video summary allows navigating in abstracted video content in a progressive manner according to the user request.

The main contribution of this work is to provide a scalable video summarization in terms of media content. This approach is inspired by hypercube On Line Analytical Processing (OLAP) operations [23]. The idea is borrowed from hierarchical information retrieval frameworks, which have become particularly popular in Multimedia archives [2]. The hypercube concept has been proposed to facilitate user's navigation through multidimensional space where each move corresponds to a query using some combination of the dimensions. In this work we consider different descriptors and embed them into consensus clustering framework which allows hypercube partitioning in multimodal audio-visual description space. To test this approach a sample of the French National Audiovisual Institute (INA) [1] cultural video corpus was used. To evaluate the performance of this method, precision and recall measures are considered. The evaluation consists of the comparison between human detection of segment boundaries (manual annotation) and the automatic visual summary obtained by the proposed method.

This paper extends former work by R. Perez-Daniel *et al.* [52] with the introduction of motion features in the cross-modal feature space, the enrichment of the corpus by 4 new videos, and the use of a complementary evaluation methodology based on reference summaries annotated by several human annotators.

1.2 Visual summaries in a cross-media space

Summarization of documentaries and other cultural programs is a challenging issue because of the absence of shot production rules. Besides, the duration, the content as well as the presentation vary a lot over each documentary. Another important aspect to consider is that the nature of a documentary does not imply several repeats of the same scene. However, frequently a documentary contains similar video content along the time and this characteristic has to be considered to build the video summary. For this reason we propose to take advantage of the multiple times that similar video content can

[1] `http://www.ina.fr`

be found in a documentary to propose a scalable video summarization approach, where each scale (level of detail) shares some attributes, with the selected keyframe of the video summary displayed allowing user's navigation in the video document.

To achieve the scalable video summarization, we follow the methodology illustrated in Figure 1, which mainly consists of two stages: *video summarization* and *scalability*.

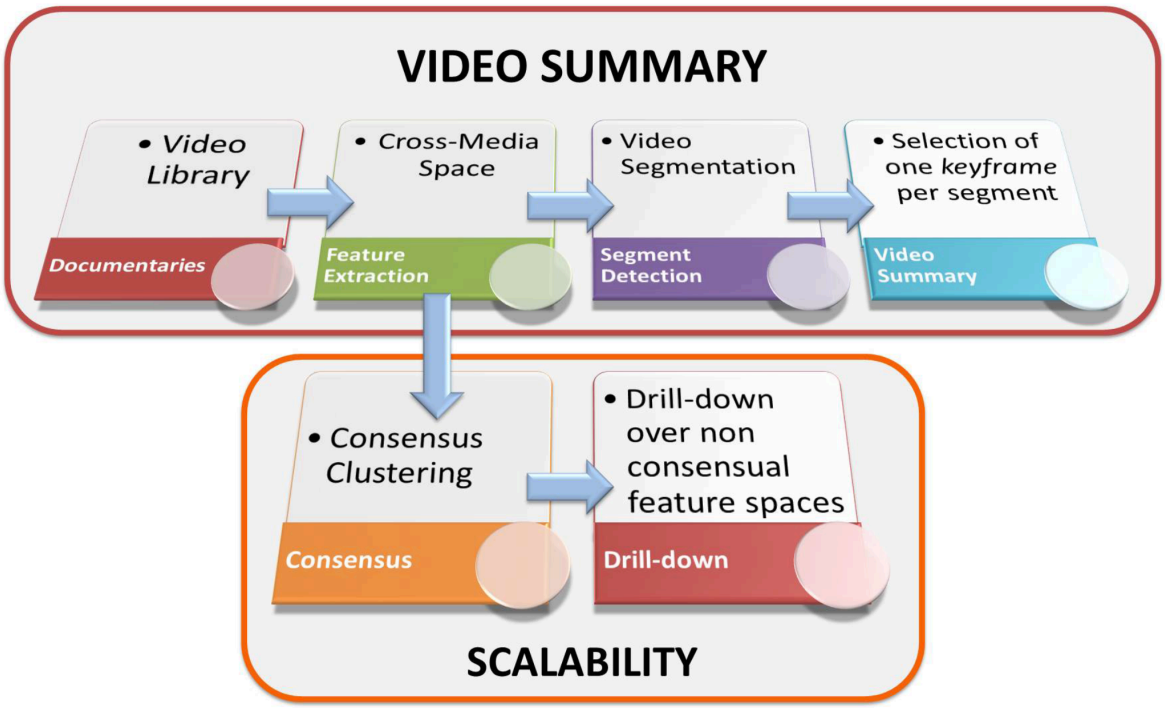

Figure 1: Proposed architecture for scalable video summarization

In the scope of this paper, the summary aimed for a video document is a compact representation consisting of a sequence of keyframes. It can be displayed as a storyboard or as a skimmed video clip. The summarization approach we propose consists of dividing the whole video into contiguous and homogeneous audio-visual segments, which are not obviously video shots: they can be longer or shorter than them.

We assume that the documentary-like nature of the videos we consider implies low redundancy in their perceptual content over time. As a consequence, we do not exploit possible similarities between the audio segments for the summarization, where a keyframe relates to a unique segment and not a class of segments.

The approach we consider for video segmentation uses the consensus clustering paradigm (see the *Consensus* block of Figure 1), in which it is possible to merge different clusterings performed over different dimensions of the description space.

Our goal is to ensure a scalable navigation in a video summary. Hence, to model a video document in a cross-media description space we use the OLAP hypercube model [23]. It allows navigating into the clusters obtained by consensus clustering according to the preferences of the user. We materialize this by the *drill down* block in Figure 1.

The following subsections present the other blocks depicted in Figure 1, which are related to feature extraction, segment detection and video summary construction.

1.2.1 Cross-media feature space

The data cube approach allows different combinations of features in the *early fusion* paradigm, often considered within multimedia document processing [51]. It is therefore necessary to express all the features, extracted from different modalities with different sampling periods, according to a similar temporal scale. To keep a high resolution for every feature, we consider the temporal scale associated to the lowest sampling period encountered (see section 1.2.1). The missing coefficients are obtained using linear interpolation.

Visual Features. To describe visual information we limit ourselves to global frame descriptors. For the color, we use the well-known MPEG7 features such as Color Structure Descriptor (CSD) [67] and Scalable Color Descriptor (SCD) [41]. CSD has presented effectiveness in image retrieval based on color [41] and in shot boundary detection [4], while SCD is more sensible to color variations. CSD [67] captures the distribution of colors in the image as well as the local spatial structure of colors. Here we have considered 64 quantization levels in HMMD color space to get a 64-dimensional vector. The second color descriptor SCD, is a histogram computed in HSV color space and then encoded by Haar transform coefficients. In this work we use 128 bins[2].

To describe the shape and texture in video frames we use the Pyramid of Histogram of Oriented Gradients (PHOG) [4] which has been proven to be efficient in recognition of global shape of objects and human actions. It represents a pyramid of blocks with the histogram of oriented gradients; we use two levels of it [64]. The dimension of this feature is 100. All visual features are computed at the frame rate of 1 fps[3].

We also consider motion features extracted from dense trajectories proposed by Wang *et al.* [63]. Their extraction consists of a classical bag of features approach learnt on feature vectors concatenating the Histogram of Oriented Gradients (HOG) of size 96, the Histogram of Optical Flow (HOF) of size 108 and the Motion Boundary Histogram (MBH, which is the gradient of optical flow) of size 96 for the x and y axis, calculated on every trajectory of every frame of the video with 25 frames per second. We consider a number of 500 motion words for computational reasons. Thus, we obtain a feature vector (histogram of size 500) per video frame[4].

Audio Features. We consider descriptions of the audio stream through Mel-Frequency Cepstral Coefficients (MFCCs) and Chroma vectors. MFCCs provide a rough description of the spectral envelope of the signal considered. Their computation relies on the discrete cosine transform of the signal's log-power spectrum, previously filtered by bandpass filters regularly spaced according to the Mel scale. The Mel scale models the logarithmic behavior of the human ear to audio stimuli. The lower MFCCs are often considered as a way to describe its overall timbre [48]. A Chroma vector is a set of coefficients which quantizes the energy associated to the twelve semi-tones of the chromatic scale in western music theory [3].

We describe the audio track through a sequence of vectors of Mel-Frequency Cepstral Coefficients (MFCCs). We extract a set of MFCCs from the temporal windows of size T centered on every multiple of T.

In this work, 13 MFCCs (including 0th order) and Chroma vectors of dimension 12 are regularly extracted from the audio using the Yaafe toolbox [40], with hop size of 2048 and 8192 points respectively, and an analysis window size of 4096 in the case of MFCCs [5]. The set of values x^d of each feature in time is then normalized by linear mapping.

Audio features are computed at higher frequency than video frame rate, therefore in this paper

[2]These features are extracted using the MPEG-7 Feature Extraction Library proposed by Bilkent University Multimedia Database Group, available at `http://cs.bilkent.edu.tr/~bilmdg/bilvideo-7/Software.html`.

[3]PHOG descriptors are computed thanks to the MATLAB script by Anna Bosch and Andrew Zisserman available at `http://www.robots.ox.ac.uk/~vgg/research/caltech/phog.html`.

[4]The extraction of HOG, HOF, MBH and the implementation of bag of words approach are realized using the Dense Trajectories Video Description Toolbox by Wang *et al.*, available at `http://lear.inrialpes.fr/people/wang/dense_trajectories`.

[5]The analysis window size is set automatically by the toolbox in the case of the Chroma vectors. The other parameters for the extraction of MFCC and Chroma vectors are set as the default ones in Yaafe.

we use the sampling rate of MFCC descriptor to generate the audio-visual description of the video by the linear interpolation of both audio and visual features.

1.2.2 Segment detection and video summary construction

In this work, we consider a video summary as a sequence of keyframes where each keyframe represents a segment of the video. A segment is defined by two time instants, and a video is entirely covered by all its segments. Each segment can be partitioned on user request to access summaries with finer levels of detail, thus ensuring the scalability property. This approach can be formalized with the data cube OLAP model.

In general terms, data cube structure [23] consists of several *dimensions*, where each dimension represents some attribute in the database, which is represented by a *measure*. In our case dimensions are given by the proposed descriptor spaces, while measures are given by the clustering of them.

Data cube offers flexibility for navigation into the data by displaying a summary at different levels of granularity. To reach this goal, data cube considers several OLAP operations such as *slice, dice, roll up and drill down*. We focus on the latter operation which implies the data summarization by climbing down hierarchically into the data. Hence in a data cube model, we need to define the clustering in a complete description space and then define it accordingly to particular dimensions. We formalize the high level clustering in a complete space as a *segment detection* and present it below.

A *segment* is a collection of data points that are close in the description space and contiguous in time. The segments of a video are obtained by applying successively a K-means clustering [25] in the feature space, then a density clustering in the temporal space. As the number of clusters K required by the K-Means algorithm is unknown, we propose to assimilate it to the target number of segments of the video summary. We therefore express K as a function of a *target summary duration* T_{summary}, where :

$$T_{\text{summary}} = \frac{T_{\text{video}} * \rho}{100} \tag{1}$$

T_{video} is the duration of the video document in seconds and ρ is the expected percentage of the summary w.r.t. the duration of the overall video document. Thus, the target number of segments is calculated as follows :

$$K = \frac{T_{\text{summary}}}{\overline{T^q_{\text{seg}}}} \tag{2}$$

where $\overline{T^q_{\text{seg}}}$ is the average duration per segment in the current category q of video, obtained from the reference annotations.

K-means offers the data partitioning according to feature similarity and often similar frames can be found in different segments along the video, which indeed can be chronologically distanced.

In pioneering works by Yeung *et al.* [66] and Benois-Pineau *et al.* [6] on visual scene segmentation, such a problem was solved by combining distances on visual features and time respectively. In the present work we aim to obtain a strict temporal connectivity of segments, using a density-based clustering [15] on temporal similarity as a post-processing stage. The density connectivity between the members of a set $S = \{s_1, s_2, ..., s_N\}$ is given by the timestamp of each member.

Let $\{S_1, S_2, ...S_K\}$ be a set of clusters obtained by K-means in the complete description space, where $S_k = \{s_{1k}, s_{2k}, ..., s_{nk}\}$, $k \in \{1, 2, ..., K\}$ and considering that s_{jk} exists at the time instant $t_{s_{jk}}$, $j \in \{1, 2, ...n_k\}$, such that $\overline{S_k} = \{t_{s_1}, t_{s_2}, ..., t_{s_n}\}$.

Then, the member $t_{s_{(j+1)}} \notin \overline{S_k}$ iff $t_{s(j+1)k} > (t_{sjk} + \tau)$. Thus, in that case, a new subset $\widehat{S}$ emerges, where τ is a temporal distance threshold (set to the highest sampling period), otherwise, the member $t_{s_{(j+1)}} \in \widehat{S_k}$. Therefore, considering the temporal distance, now the set of partitions is given by $\widehat{S} = \{\widehat{S_1}, \widehat{S_2}, ..., \widehat{S_Q}\}$.

Finally the set of partitions $\{\widehat{S_1}, \widehat{S_2}, ..., \widehat{S_Q}\}$ is chronologically sorted to represent the chronological occurrence of each segment as $\widehat{S_p}$.

1.3 Scalable video summary navigation based on data cube and consensus clustering

Scalable video summary in terms of content description refers to the multidimensional access to different feature spaces. Scalability makes it possible to navigate in the video summary to get detailed information about the selected segment, over the selected feature space.

Consensus clustering is the process of merging multiple clusterings performed on the same dataset with different parameters [19]. In the case of our cross-modal description space, we propose to build several partitions of the same dataset, which is one video document. The process presented in section 1.2 is repeated on different subspaces of a complete description space yielding a pre-computed set of partitions of the same dataset. We propose the scalable navigation in the dataset using less consensual partitions of clusters. We introduce this formally in the following.

Let us define a video V as a finite dataset of cardinality $V = m$, which is described by η feature spaces. From the clustering of each feature space we can get a set ν of η set partitions ϕ such that $\nu = \{\phi_1, \phi_2, ..., \phi_\eta\}$, where ϕ_i is a set partition of the i-th feature space.

Agreements and disagreements between two partitions ϕ_i and ϕ_j of ν can be calculated by considering all pairs of elements in ϕ_i and ϕ_j, where an *agreement* is given by $a_{ij_1} = \{$pairs that are co-clustered in ϕ_i and $\phi_j\}$ and $a_{ij_2} = \{$pairs that are neither co-clustered in ϕ_i nor $\phi_j\}$, while a *disagreement* is given by $b_{ij_1} = \{$pairs that are co-clustered in ϕ_i but not in $\phi_j\}$ and $b_{ij_2} = \{$pairs that are co-clustered in ϕ_j but not in $\phi_i\}$.

Symmetric distance difference (sdd) can be used to measure the distance d between two partitions according to equation (3) :

$$d(\phi_i, \phi_j) = b_{ij_1} + b_{ij_2} = \binom{m}{2} - (a_{ij_1} + a_{ij_2}) \tag{3}$$

where $\binom{m}{2}$ are the pairs of members of clusters. To compare the distance between partitions in the set of partitions $\{\phi_1, \phi_2, ..., \phi_\eta\}$, let us consider the sum of distances SD, defined as:

$$\mathrm{SD} = \sum_{i=1}^{\eta} d(\phi_i, \phi_j) \tag{4}$$

where η is the number of set partitions, ϕ_i is the current set partition and ϕ_j is a specified partition, where $\phi_j \in \nu$. Thus the most consensual set partition is given by the one with the minimum value of SD, while the least consensual one is defined by the set partition with the maximum value of SD.

Feature spaces with lower consensus are indeed, different views of the video summary but still they have some similar visual content. Hence, they can be used within the drill down of the data cube. It means a selection of a pre-computed partition of a cluster of interest in the less consensual subspace. An example is shown in Figure 2 where the three less consensual subspaces are considered. The dimensions of the data cube are A, B and C, where A, B, C are the three feature spaces with less consensus and the measures are clusters over dimensions, where the number of clusters is given by equation (2). The base cuboid is yielded by clustering over ABC. The cuboids of the first level are the clusters obtained by considering AB, AC and BC dimensions combinations respectively, while the cuboids of the second level are the clusters regarding each single feature.

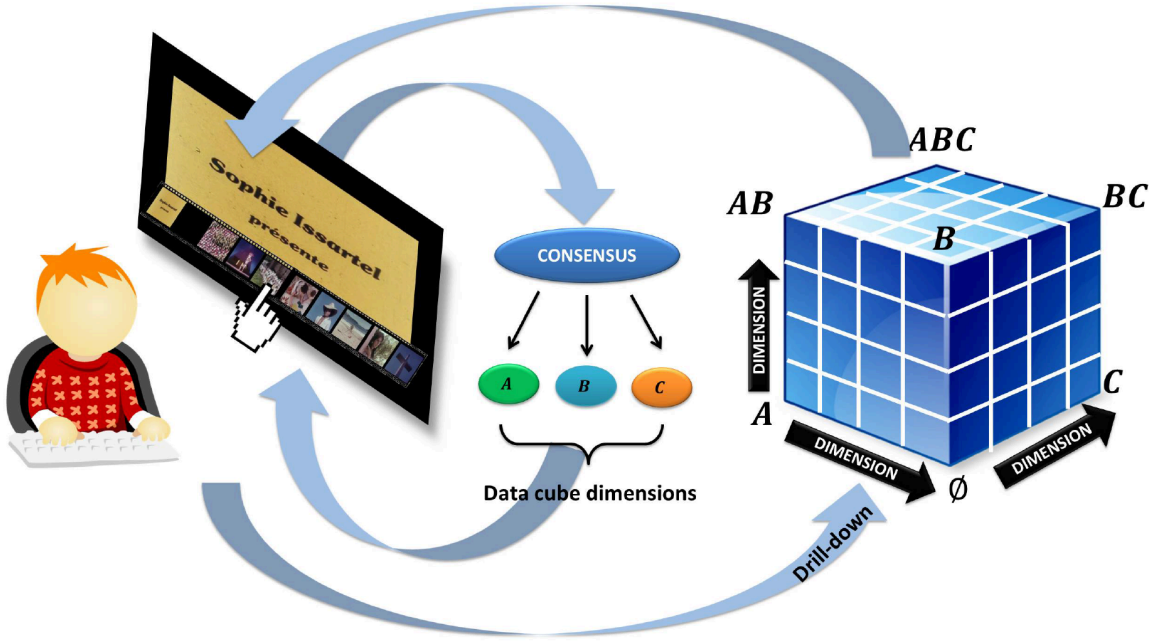

Figure 2: Data cube and user interaction

In a scalable video summary paradigm (Figure 2), the user refines the current summary by selecting one of its keyframes with respect to a particular subspace of the cross-media feature space. He or she accesses the summary of the associated segment, *i.e* a new sequence of keyframes. In practice, these keyframes are the median frames of the sub-segments obtained by the segmentation method described in section 1.2.2 according to the selected subspace.

1.4 Evaluation and discussion

In this section are presented the corpus and the metrics considered for evaluation of our approach. Then, the results are discussed regarding two basic baseline systems.

1.4.1 Corpus

The performances of the proposed method in terms of summarization quality were calculated on a sample of a corpus of video archives provided by the french National Audiovisual Institute INA. The whole corpus consists of a dataset of over 1250 french TV programs which includes documentaries, broadcast news, TV shows, musical comedies and dancing performances, among others. It is organized in 9 subsets of videos according to the keyword used for their retrieval within the INA browsing system. These keywords are mainly related to Mexican culture, due to the scope of the Mexculture project.

The sample of the INA corpus we consider for evaluation is a group of 18 videos listed in table 1, covering the 9 subsets mentioned. This corresponds to a total of 12 hours, 27 minutes and 59 seconds of video. The average duration of a video is 41 minutes and 33 seconds.

The evaluation of a video summary is a difficult issue [36], as there's not a unique "good" summary. As the annotation of a corpus is costly in terms of time and human workforce, two sets of ground truth annotation were produced. In the first one, A_1, a single annotator produced a summary for the 18 videos of the corpus. In the second one noted A_2, 6 annotators produced the annotation of 16 over the 18 videos. (The missing videos correspond to indexes 1 and 3 in Table 1). The datasets used by Li and Merialdo [36] were not considered in this work for copyright reasons.

Table 1: Subset of the INA corpus of french TV program archives.

Index	Semantic class	name
1	Corrida	FPVDB03081308_VIS_01
2	Corrida	FPVDB06122105_VIS_01
3	Corrida	KPCAB890307_VIS_04
4	Corrida	MGAFE0130859–AM_VIS_01
5	Danse	MGCPC0185354–AM_VIS_01
6	Danse	MGCPC0185354–AM_VIS_02
7	Danse	MGCPF0063313–AM_VIS_01
8	Danse	MGCPA0025472_VIS_01
9	Danse	KMCPC89081901_VIS_01
10	Danse	MGCPC0097939–AL_VIS_01
11	Danse_folklorique	FPVDB04032511_VIS_01
12	Mariachis	FMVDD092596-AFP_VIS_01
13	Mariachis	MGCPC0034156_VIS_01
14	Mexique_musique	MGAFE460101B5601_VIS_01
15	Norteno	MGCPF0105773–AK_VIS_01
16	Paysages_mexique	MGCPB0051203–AA_VIS_01
17	Site_archeologique_mexique	MGCPC0284691_VIS_01
18	Thalassa	MGCAC0018014–BM_VIS_01

Figure 3: Thumbnails for the 18 videos of the corpus used for evaluation. The numbers correspond to the video indexes in Table 1

1.4.2 Evaluation metrics

We first consider classic precision P and recall R measures. They are defined as:

$$P = \frac{\mathcal{TP}}{\mathcal{TP} + \mathcal{FP}} \tag{5}$$

$$R = \frac{\mathcal{TP}}{\mathcal{TP} + \mathcal{FN}} \tag{6}$$

$\mathcal{TP}$ is the number of true positives, and $\mathcal{FP}$ is the number of false positives. In the context of summary evaluation, we consider that a true positive is counted when a keyframe of the computed summary is contained within a segment of the ground truth, and is the first keyframe from the summary to be contained within this segment. If it is not the first keyframe occurring in this segment, it is counted as a false positive. $\mathcal{FN}$, the number of false negatives, is obtained by counting the number of segments from the ground truth which do not contain any keyframe from the computed summary.

The second set of metrics we consider correspond to Precision and Recall within the VERT framework [36]. They are inspired from evaluation protocols proposed in the text processing domain that take into account multiple ground truths. They are defined as follows. Let $\{f_m^A\}$ be the sequence of M keyframes of the summary produced automatically, and let $\{f_n^{R_q}\}$ be the sequence of N_q keyframes from the reference summary annotated by annotator q. We note Q the number of annotators. We attribute to every frame of the video a particular weight that we store in vector W^A for the automatic summary, and W^{R_q} for the reference summary by the q^{th} annotator. If the current frame f_i is a keyframe, $1 \leq i \leq I$ with I the number of frames of the current video, then its weight equals 1, otherwise its weight equals 0. We note $W_{\max}$ the vector containing the maximal weight associated to every frame by an annotator, so as $W_{\max}(f_i) = \max_q\{W^{R_q}(f_i)\}$. The VERT Precision and Recall metric are respectively defined as:

$$P_{\text{VERT}} = \frac{\sum_{i=1}^{I} \min\left[W^A(f_i), W_{\max}(f_i)\right]}{\sum_{i=1}^{m} W^A(f_i)} \tag{7}$$

and

$$R_{\text{VERT}} = \frac{\sum_{q=1}^{Q} \sum_{n=1}^{n_q} W^A(f_i^{R_q})}{\sum_{q=1}^{Q} \sum_{n=1}^{n_q} W^{R_q}(f_i^{R_q})} \tag{8}$$

In this paper, R_{VERT} corresponds to the R_1 metric defined in [36]. R_2 metrics are left aside because of the assumed temporal independence of the keyframes of a summary. Besides, such definitions imply an exact match between the keyframes. As the location of the keyframe which represents a segment is arbitrary, we propose to incorporate a temporal tolerance within the calculation of P_{VERT} and R_{VERT}. This is done through the modification of W^A and $\{W_q^R\}_q$, by attributing a weight of 1 to the frames contained within a temporal window of length $2tol + 1$ frames centered on every keyframe from the automatic and the q^{th} reference summaries respectively. Parameter tol represents the half length of the tolerance window.

1.4.3 Evaluation

The scalable video summarization model relies on the idea that the refined version of the summary should contain more fine segmentation inside a cluster obtained in the data-cube computation in the complete space.

We propose to account for refinements of the summaries within the evaluation by considering the following process:

1. Select several keyframes of the summary.
2. *drill down* over the associated segments, i.e. produce the summary for the segment related to these keyframes.
3. Compare the refined summary (global summary with the keyframes from step 1. refined by the summaries produced in step 2.) to the reference summary.

To assess the scalable video summary, 10 segments of the early-fused segmentation were randomly selected to drill down over them. We assume that this number reflects the maximum number of clicks (or segment selections) a user would do to browse a single video before inspecting another one. ρ is set to the typical value of 10%.

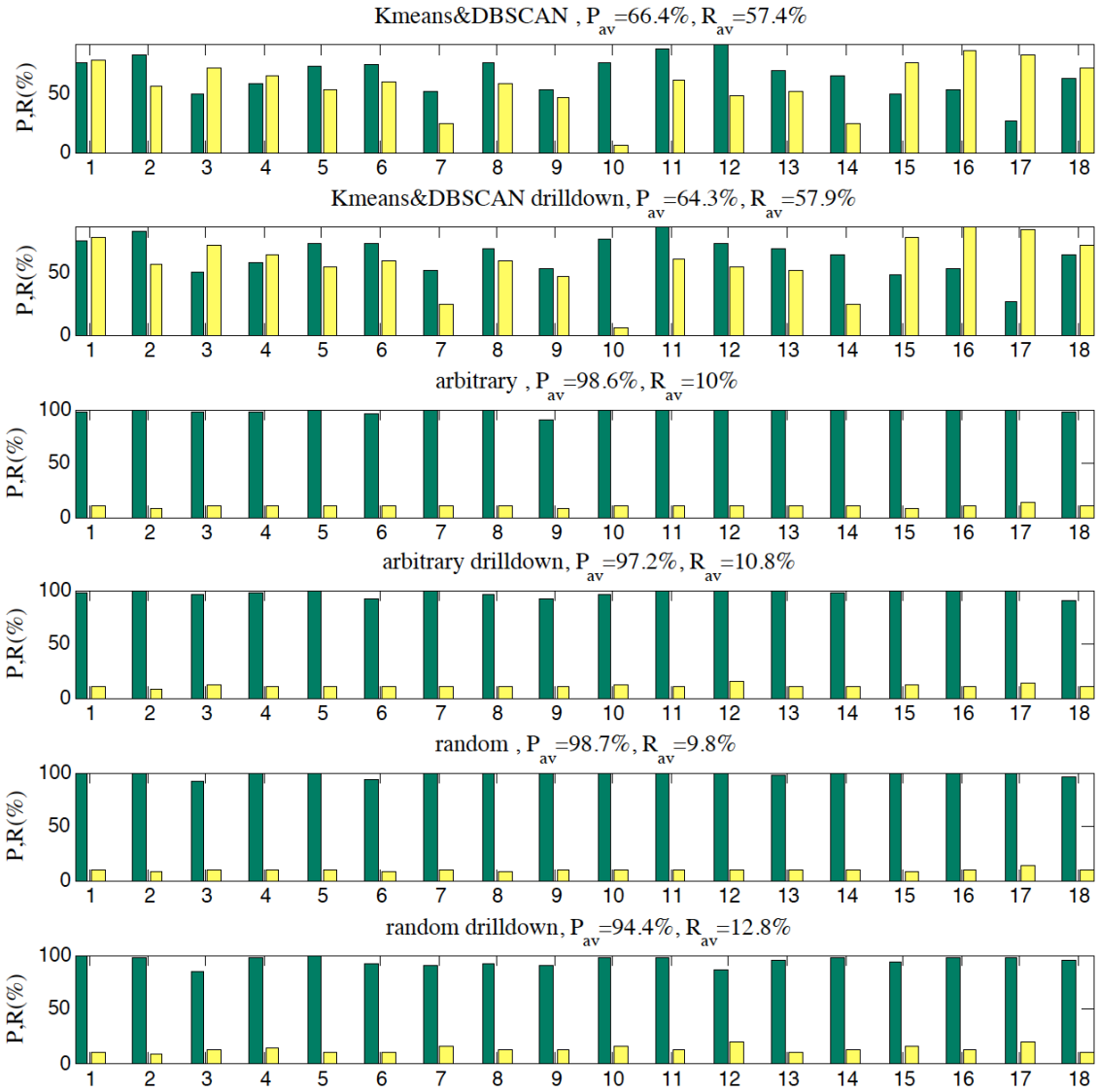

Figure 4: Precision (green bars) and recall (yellow bars) with and without drill down over ten randomly selected segments w.r.t. the reference summaries of A_1. Two additional segmentation processes are considered to position our approach. The average values of precision P_{av} and recall R_{av} over the videos are given for each case. Parameter ρ is arbitrarily set to 10%.

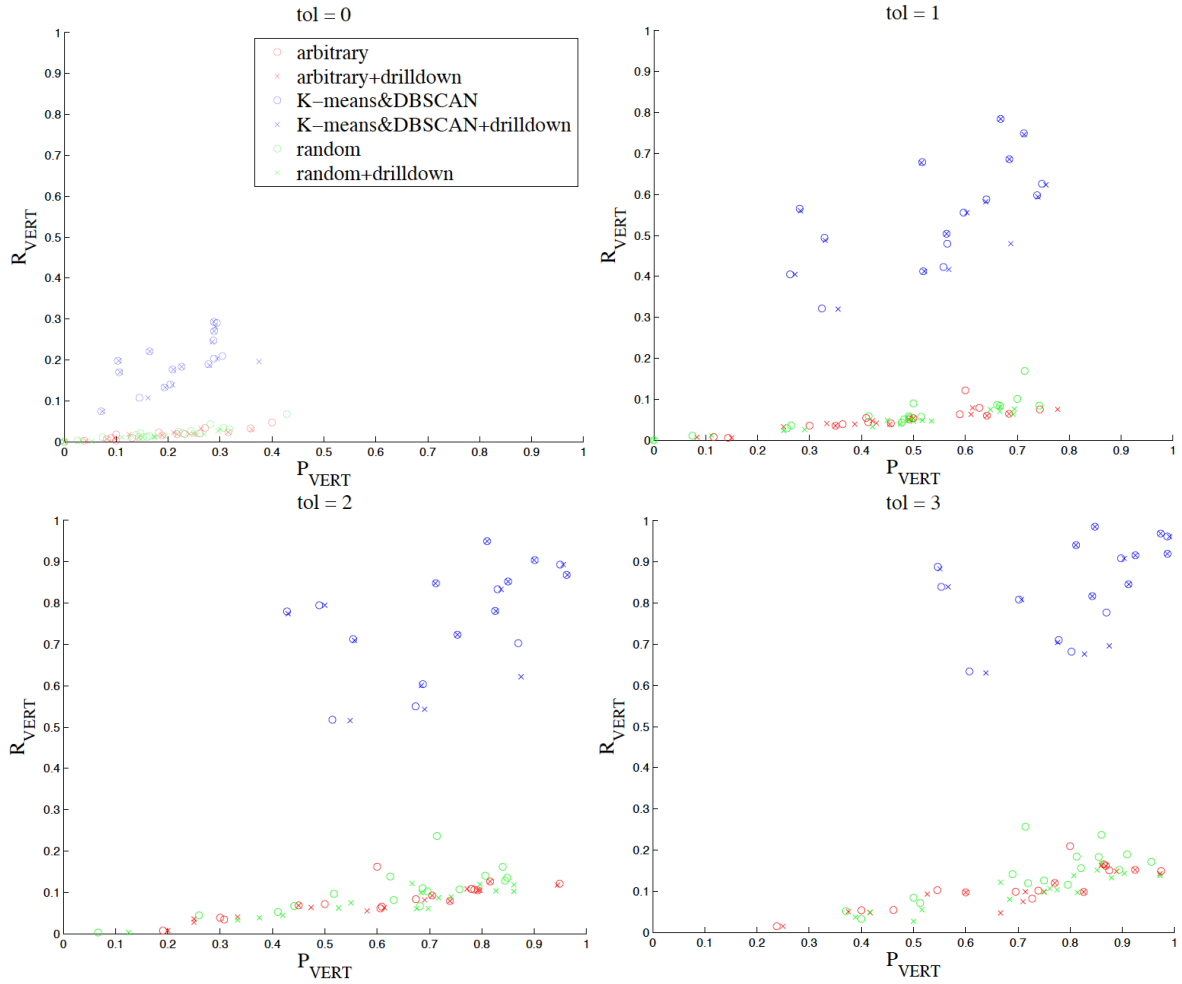

Figure 5: Values of P_{VERT} and R_{VERT} obtained on A_2 for the three summary systems, with (cross) and without (circle) the drill downs on 10 random segments. Each graph corresponds to one of the four values of temporal tolerance: $tol = \{0, 1, 2, 3\}$. Parameter ρ is arbitrarily set to 10%.

Figures 4 and 5 show the performances of our approach ("Kmeans&DBCSAN"), with and without drill downs, for the set of annotations A_1 (with classical metrics P and R) and A_2 (VERT metrics P_{VERT} R_{VERT}) respectively. Parameter ρ is arbitrarily set to 10%. To position our approach, two baseline summarization systems performing basic segmentations were considered: one segmenting the video randomly, and one segmenting the video in K segments of equal length, with K obtained according to equation 2.

Figure 4 shows that our method produces balanced values of precision and recall for most videos, whereas baseline systems give very high precision and very low recall. The drill down was realized for 10 segments from the segmentation based on early-fused features, which were sub-segmented according to one of the two features associated to the two less consensual clusterings, selected randomly. The number of clusters K' used for the clustering of the segment selected for drill down is proportional to the length of the segment.

$$K' = \frac{K * T_{\text{seg}}}{T_{\text{video}}} \tag{9}$$

with T_{seg} the duration of the targeted segment.

One can observe that the drill down step does not affect significantly the results. In the case our approach, the average precision decreases a little when the average recall increases very slightly. Indeed, the drill down involves an increase in the number of keyframes selected for the summary, resulting in a highest probability that a keyframe is counted as a false negative.

Figure 5 shows the results of the VERT evaluation on A_2 for four values of tolerance: 0,1,2 and 3, corresponding to an exact matching and tolerance windows of 3, 5 and 7 seconds respectively. Here again, we observe that values of P_{VERT} and R_{VERT} for our approach are close to the main diagonal, i.e. balanced. This is not the case for the two baselines, which perform poorly in terms of R_{VERT}. The crosses remain close to the circles, showing little effect due to the drill down even if it increases slightly the precision for several videos.

These results show that the summaries obtained with our approach are more comparable to summaries annotated by humans than the ones obtained with the two baseline systems.

We tested the drill down operation for data cube navigating into the clusters of the less consensual feature spaces according to the selected keyframe. Due to pre-computed data cube construction, the data cube navigation by OLAP drill down operation to get the refined version of the summary is less than 1 second in a Intel Core 2 Duo 2.53 GHz processor with 4 GB of RAM.

1.5 Conclusion

In this paper, we have presented a method for construction of scalable video summaries of audio-visual documents based on data cube architecture. This approach provides a customized access to different versions of different levels of detail of a video summary in cross media space. The proposed video summary relies on nonconsensual feature spaces to achieve scalability. We have performed an evaluation of the proposed method with regard to video summaries obtained by a random selection of clusters and arbitrary abstraction with a constant time step and summaries obtained from humans. The method was applied to generic video content without a clearly defined structure, such as cultural documentaries. At this stage of research it is difficult to assess the completeness of the proposed summary with regard to user requirements. Indeed the user discovers the content via scalable browsing. Hence a very large-scale experiment is needed for such an assessment, which has to be conducted in trials following this research work.

In the perspective of this work we will consider richer description spaces, investigate the benefits of alternative and complementary clusterings in the subspace selection, work out the user navigation interface and perform experiments at a large scale.

As another direction for future work, we intend to investigate the partial materialization of the whole data cube. Indeed, the size of the latter may become too large due to the exponential number of cuboids. Each of which can be so large that a full materialization may become unfeasible in practice. The solution then would be to select just the most *beneficial* cuboids or levels depending on what we consider as beneficial. The remaining cuboids/levels will be computed online.

2 Scalable Action Detection and Retrieval

2.1 Introduction

Digitization of video archives is a conservation measure allowing to preserve historical and cultural video productions that mitigates the risk of loss due to the volatility of physical media. Digitization also creates the possibility of wide public access to such content permitting browsing, discovery and specific queries through Web interfaces. Human actions contain important information about the cultural content of video archives for research or educational use. Moreover, automatic detection of such actions has a wide range of applications in video surveillance, monitoring of patients and Human-Computer interaction. In this work we aim to perform retrieval of human actions in large digitized video archives.

Localization of human actions in video answers two questions important to a user: "does the action of interest occur in the video? if so then where and when?". Thus the problem first involves a detection step: determining if the database contains any instances of the action that are similar to the query examples. Next, a localization step is needed: finding, in the video timeline, the bounds of the action example that separate it from non-relevant content. Detection and localization of human actions in videos is challenging because of the complexity and variability of human motions, but also because of the large amount of video data to be searched. It remains an open problem, despite intensive research during the past decade. In this work we are not only interested in action localization in short videos, but in building a system that achieves this in a reasonable time when the collection can contain hundreds of hours of video.

The definition of actions in general can be broad but three semantic levels are typically considered. An *atomic* action is simply a short coherent elementary movement such as "raise hands", "lower hands" or "move leg". At this level, research mainly focuses on modeling such actions as statistical processes [24] or as time series [72]. At the *intermediate* level, an action is composed of a series of atomic parts and can vary in complexity, *e.g.* from "smoking" to "pole vaulting". Most of the recent research considers this level and the focus is on finding ways to aggregate atomic descriptions. Finally, at a *higher* semantic level, interest is in "events" that group actions into classes having high variability in terms of both atomic components and temporal organization (*e.g.* "making a sandwich" in the TRECVID MED challenge [46]). Complex background, variability in point of view, occlusions and low video quality pose a challenge for action detection in video.

We aim to perform action-based indexing of large scale cultural video databases in order to support broad access to such content. This paper is an extension of the short paper presented at the IEEE IWCIM 2014 workshop [58]. We focus on scalable retrieval and localization of *intermediate* level actions where the query is given as a set of action examples (the positive class). We make four contributions that distinguish our system from previous ones:

1. To take advantage of the temporal information, we represent actions as time series and compare them using the Global Alignment (GA) kernel,

2. To find a better balance between efficiency end effectiveness, we propose a cascaded approach that employs both aggregated and frame level information,

3. To improve time series comparisons with the GA kernel, we introduce a novel feature selection method for sparse multivariate time series,

4. We introduce a novel large scale localization data set that presents new challenges.

The system presented here is evaluated on *MEXaction*, a novel large scale cultural data set we created by annotating videos from the archives of the Institut National de l'Audiovisuel (France). The collection is described in Section 2.4.1. We compare our approach to state-of-the art methods, both on the MEXaction data set and on existing benchmarks featuring actions that are similar to the ones we aim to find: Smoking and Drinking [35] and MSR Action II[6] [71].

In Subsection 2.2 we discuss related work and briefly introduce the tools we employ: "tracklet" descriptors and the Global Alignment kernel. In Subsection 2.3 we describe our cascaded approach and in Subsection 2.4 we introduce the MEXaction data set and present the experimental validation of our system. We conclude and discuss the perspectives of this work in Subsection 2.5.

2.2 Related Work

Actions in video are modeled using either global descriptions of spatio-temporal volumes of the video or sets of local features describing spatio-temporal patches. With local features, modeling relies on their statistical distribution over a volume of the video.

[6] http://research.microsoft.com/en-us/um/people/zliu/ActionRecoRsrc/

2.2.1 Holistic methods

Global (or "holistic") methods usually aim to locate a human performing a certain action captured in video under controlled conditions. Either the action is described using a volumetric descriptor (2D+time) or the human silhouette is extracted and its shape described by a parametric model [7].

Laptev and Pérez [35] proposed a sliding volume approach for action detection and localization in both space and time with SVM classifiers. Videos are described by discretized orientation histograms of dense optical flow extracted from each volume. Detection is based on a boosted cascaded approach using the framework of Viola and Jones [61]. Furthermore, the authors first filter the video using still-image based object detectors trained to produce candidate video sequences for the motion-based classifiers. Yeo et al. [65] compare the motion field of query clips to target video volumes to localize actions in space and time. The motion is estimated using compressed domain information in order to speed up single example queries for simple actions.

The use of Motion History Images (MHI), a global image-based method relying on silhouettes, [12], remains a popular approach. In an MHI a pixel is assigned a value representing the temporal history of the motion at its spatial coordinates. Action recognition is performed with a nearest neighbor classifier using the Mahalanobis distance between the moments of the example and query MHIs. More recently, Tian et al. [59] combine MHI and local features to localize actions on the MSR Action II data set.

Volumetric methods are particularly useful for precise spatio-temporal localization but expensive when used at a large scale. Moreover, invariance to action length requires testing video volumes of different lengths, slowing down the system.

2.2.2 Local feature statistics

Methods based on local features recognize actions through the statistics of sets (bags) of descriptors of small video regions, not necessarily linked to body parts or image coordinates. The advantage is in avoiding the segmentation of the human from the background, which is prone to errors. The computation of a costly description of a full video volume is not needed neither.

Local features describing the dynamics of video patches were first defined as extensions of image interest points (*e.g.* Harris and SIFT points) to the spatio-temporal domain, giving Space Time Interest Points (STIPs). Laptev [34] couples such an interest point detector based on spatio-temporal scale space with a gradient and optical flow description of the patches around these points. An improved detector of such points using Gabor filter responses was proposed by Dollar et al. [13]. More recent trajectory-based features (Raptis and Soatto [53]) aim to describe both the trajectories and the spatial neighborhoods of salient points. These descriptions include local shape (histogram of gradients, HoG), optical flow (histogram of optical flow, HoF) and optical flow gradient (motion boundary histogram, MBH, by Dalal et al. [10]).

The statistical distribution of the local features for each frame is usually represented by a Bag of Visual Words (BoVW) histogram, quantizing feature descriptors to visual words and counting their occurrences in a video sequence. The BoVW of a video segment containing multiple frames can be obtained by aggregation. A simple method consists in summing up the consecutive frame histograms.

Gaidon et al. [17] perform temporal localization by using a sequential model of the action volume in which a soft ordering between "meaningful temporal parts" is imposed. Frames are grouped into "actoms" that are subsequences described by an aggregation of their underlying BoVW of STIP features. An action is a sequence of a fixed number (3) of actoms with varying temporal extents. Detection is done with an SVM classifier testing several potential aggregations of the BoVW sequences. The main difficulty with this method is the need for actom annotations in order to learn the temporal models of the actions. Klaser et al. [32] proposed a two stage approach for spatio-temporal localization of human actions. First, a person detector (using a fast linear SVM) allows to filter out uninteresting windows. Second, an action detector is learned using HoG-Track descriptions and applied to improve the detection performance of the first stage. The advantage of their method is in

a precise frame-level annotation of the actions through the video in space and time. Oneata et al. [44] achieve state of the art results for temporal localization by using Fisher Vectors to describe the distribution of trajectory features per frame. While BoVW histograms only capture 0-th order statistics (counts) of the local features, Fisher Vectors capture 1-st and 2-nd order statistics.Oneata et al. [45] further improve this method by using Branch and Bound (B&B) to refine the locations of detected actions. These approaches deal with interesting actions but only results on small data sets have been presented. Furthermore, the evaluation seldom considers the analysis of an action localization system as a whole.

Spatio-temporal action localization was also formulated as the problem of finding the most discriminative video sub-volume with respect to a score function computed on the features contained in the volume. Scoring is based on the mutual information between the bag of STIP features in the sub-volume and the training set features, thus requiring the computation of nearest neighbors in feature space. Yuan et al. [71] describe the basic method, using Branch and Bound (B&B) search to locate action volumes, iterating through all interesting detection volumes one by one. To speed up search, in Yuan et al. [70] the authors decouple the space and time dimensions for B&B search. Goussies et al. [22] optimize the search by finding all actions in a single round of search. Finally, Yu et al. [68] propose a fast random forest scoring method that removes the need of the costly nearest neighbor search. Yu et al. [69] discretize the score search space and replace B&B with Hough voting and Maximum Subarray search obtaining a large increase in action retrieval speed. Oikonomopoulos et al. [43] use the positions of characteristic ensembles of feature descriptors to localize actions. For each class, local feature ensembles are found through feature selection. Ning et al. [42] use biological features and propose a 5-stage coarse-to-fine model in which various aggregations of low level features permit filtering of non-relevant test video volumes. While achieving good localization performance, these works deal only with simple actions with low degrees of variability in small video collections, mostly aiming at real-time automatic surveillance.

Shao et al. [56] and Jones et al. [29] introduce a relevance feedback-based system for action localization and retrieval. In a sliding window approach, they use the asymmetric bagging SVM classifier that mitigates the problem of small numbers of action examples. Starting with a single query video sequence and building the query model using relevance feedback, they perform action localization using B&B search and candidate re-ranking. Even though imbalance is also an issue in our case, methods based on bagging and ensembles or methods that deal with the class imbalance by densifying the examples may not be the best choice for a large scale setting because of their high computation cost.

2.2.3 Temporal matching

As shown in 2.2.2, most methods for action localization rely either on the temporal aggregation of BoVW histograms or on scoring sets of local features in spatio-temporal volumes. However, temporal averaging loses potentially important information regarding both relative durations of action parts and their temporal ordering. In Gaidon et al. [18] videos are represented as high-dimensional time series (each frame described by its BoVW) and the autocorrelation of each time series is used as a compact description of its dynamic behavior. While reflecting the temporal evolution to some extent, autocorrelations lack discriminative power if temporal deformations, e.g. faster/slower action speed, are frequent.

Dynamic Time Warping [54] (DTW) is a method for matching two time series that takes temporal ordering into account but is tolerant to temporal deformations. Consider two time series, $\mathbf{Q}$ and $\mathbf{X}$, each being an ordered list $\mathbf{Q} = \mathbf{q}_1, \mathbf{q}_2, ...\mathbf{q}_M$, $\mathbf{X} = \mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_N$ of d-dimensional vectors $\mathbf{q}_i$ and $\mathbf{x}_j$. DTW finds the best warping path $W^* \in \mathcal{A}$ (the set of all possible warping paths), $W^* = w_1, w_2, ..., w_K$, with $w_k = (i, j)$, between $\mathbf{Q}$ and $\mathbf{X}$ such that the total cost of the match, C_{W^*}, is

minimized, with respect to a cost function c.

$$DTW(Q, X) = C_{W^*} = \min_W C_W = \min_W \sum_{k=1}^{|W|} c(w_k) \tag{10}$$

The cost function is a dissimilarity measure such as χ^2 or, more often, a distance such as L_2:

$$c(w_k) = c(\mathbf{q_i}, \mathbf{x_j}) = \|\mathbf{q_i} - \mathbf{x_j}\|_2 = \sqrt{\sum_{k=1}^{d} (\mathbf{q_{ik}} - \mathbf{x_{jk}})^2} \tag{11}$$

Clearly, such an approach is computationally more expensive than simple frame aggregation even when employing dynamic programming for its calculation. When describing frames using d-dimensional BoVW, if both video sequences have equal length L, the complexity of testing all possible warping paths is $O(dL^2)$. In comparison, the complexity of matching with aggregation is $O(d)$ (does not depend on L).

DTW cannot be directly employed to build a positive definite kernel for time series. Cuturi [9] replaces the *min* calculation with a *soft-minimum* (eq. 12) of all the path costs. They show that the exponentiated *soft-minimum* leads to a positive definite kernel, the Global Alignment (GA) kernel k_{GA} (eq. 15), when pairwise comparison between frames (eq. 14) uses a kernel derived from the Gaussian kernel κ_σ (eq. 13).

$$\text{softmin}(C_{\mathcal{A}}) = \log \left(\sum_{W \in \mathcal{A}} exp(-C_W) \right) \tag{12}$$

$$\kappa_\sigma(\mathbf{q_i}, \mathbf{x_j}) = \exp\left(-\frac{\|\mathbf{q_i} - \mathbf{x_j}\|_2^2}{2\sigma^2} \right) \tag{13}$$

$$c(w_k) = c(\mathbf{q_i}, \mathbf{x_j}) = \frac{\kappa_\sigma(\mathbf{q_i}, \mathbf{x_j})}{2 - \kappa_\sigma(\mathbf{q_i}, \mathbf{x_j})} \tag{14}$$

$$k_{GA}(\mathbf{Q}, \mathbf{X}) = \exp(\text{softmin}(C_{\mathcal{A}})) = \sum_{W \in \mathcal{A}} \exp\left\{ -\sum_{k=1}^{|W|} c(w_k) \right\} \tag{15}$$

2.3 Methodology

We propose here a two stage cascaded approach to temporal action detection and localization, exploiting the complementarity between temporal aggregation at the first stage and video frame alignment at the second stage. Detection is performed using a sliding window and a cascade of SVM classifiers, while a post-processing step gives the temporal localization of actions. Fig. 6 shows the block diagram of this method.

2.3.1 Video time series description

To describe video content for action detection and localization we follow [62]: points are sampled on a regular grid in each frame and tracked across 15 frames. Tracking is done by motion estimation between consecutive frames, based on the dense optical flow estimated with the method from [16]. The trajectory of a point consists of the coordinates of the point in consecutive frames. A "tracklet" is defined as the concatenation of three features, HoG, HoF and MBH, computed in patches around the trajectory points, which results in 396-dimensional descriptors. We quantize these descriptors using K-means into a visual dictionary $\mathcal{W}$ of $d = 4000$ words and compute a BoVW histogram for

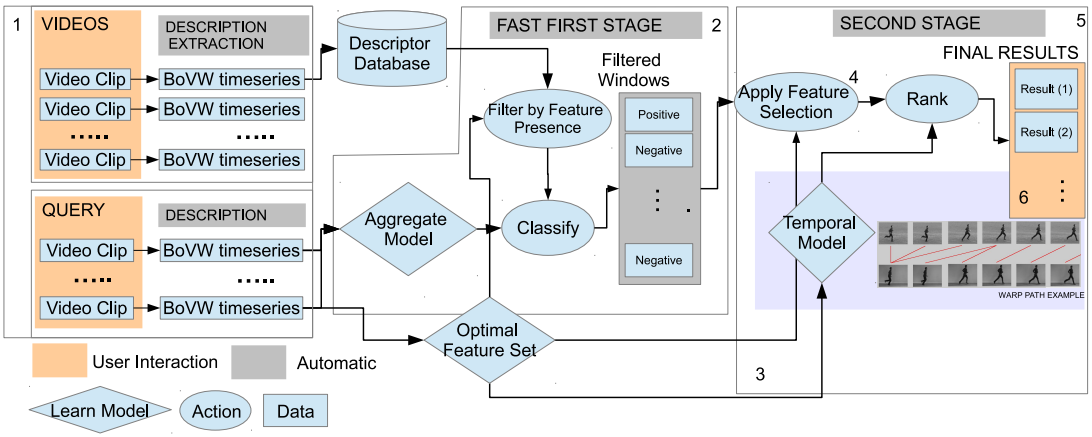

Figure 6: Block diagram of the proposed method.

each frame. These sparse vectors, one for each video frame, constitute a high-dimensional time series description of the video. In the following, *sequence* refers to any segment, or sub-series, of a tracklet BoVW time series. A *window* refers to a sequence of $L = 30$ frames. In Fig. 6 the description process is illustrated in block 1. These particular values for parameters d and L were shown to give good performance in [35],[17].

2.3.2 Training approach for the two stages

The cascade uses two classifiers, one for each stage, in a One-vs-All discriminative setting. This section describes the approach used to produce the training data and applies to both classifiers. For an action class, positives (action examples) are obtained by extracting the sequences spanned by ground truth annotations. When the annotations in the ground truth refer to restricted spatio-temporal volumes (not covering entire frames), we extract BoVW histograms only from these volumes and use them as positive examples. Since ground truth annotations are often imprecise, we add as positive examples data extracted from jittered ground truth windows with a temporal amplitude of $\pm$ 10 frames and, when possible, a spatial amplitude of $\pm$ 25% of the window size. The BoVW time series in these sequences are subsampled to be $L = 30$ frames long.

Since ground truth positive examples can be long, to allow for sub-sequence matching we also add as positives all L length windows sampled at regular time intervals that overlap the ground truth windows. If a training video is just a long positive example (as for the KTH data set, see Section 2.4), this fixed stride sampling provides the positive training windows.

For negative (non-action) examples we sample windows from the training videos, choosing among those that do not contain any annotated examples of the positive class.

2.3.3 Cascade First Stage

The first stage of the cascade has to filter out a maximum of windows that are very unlikely to contain the action of interest. It is directly applied on all the windows extracted from the videos in which we aim to locate actions (block 2 in Fig. 6). At this level, a window is described by the renormalized sum of the BoVW histograms of all its frames (denoted below $X^{(agg)}$). The first stage classifier has to to decide whether a window is relevant (to be sent to the second level) or should be filtered out. This decision is taken according to the value of the decision function of an SVM classifier: relevant iff $f_1(X^{(agg)}) \geq \tau_1$. For the first stage, we use 2-class SVMs with the Histogram Intersection (HI)

kernel:

$$k_{HI}(Q, X) = \sum_{i=1}^{|\mathcal{W}|} min(Q_i^{(agg)}, X_i^{(agg)}) \tag{16}$$

Given a set of aggregated sequences $\mathcal{X}$, the first stage should filter out a maximum of the irrelevant windows in order to reduce overall detection cost, while keeping recall high. The number of positives $\mathcal{X}^+ = \{X \in \mathcal{X} | class(X) = Positive\}$ that pass this stage should be as high as possible, thus minimizing the false negative rate (FNR_1).

$$FNR_1 = \frac{Card(\{X^{(agg)} \in \mathcal{X}^+ | f_1(X^{(agg)}) < \tau_1\})}{Card(\mathcal{X}^+)} \tag{17}$$

We denote by *coverage* the ratio of windows found relevant at the first stage and that will be ranked at the second stage (i.e. the positive rate).

$$Coverage = \frac{Card(\{X^{(agg)} \in \mathcal{X} | f_1(X^{(agg)}) \geq \tau_1\})}{Card(\mathcal{X})} \tag{18}$$

The τ_1 parameter controls the trade-off between the false negative rate and the coverage: the former increases while the latter decreases with τ_1. We thus seek C, the 2-class SVM regularization parameter, and τ_1 for the first stage classifier to minimize $\alpha FNR_1 + (1-\alpha)Coverage$, where $\alpha = 0.8$ to favor the minimization of the FNR_1.

2.3.4 Cascade Second Stage

At the second stage, windows are classified as Positive or Negative and ranked according to a score measuring similarity to the query model. For model construction, this stage combines selecting good features for temporal alignment of frames with learning an SVM model using the Global Alignment kernel. For prediction we combine pruning candidate sequences based on the presence of selected features with scoring based on the SVM decision value.

Feature Selection. In a large video collection the action of interest will appear rarely. Moreover, the action does not always cover the whole frame but can be localized in a region of the image plane. We call "background" all the information that is not localized within the extents of the actions of interest. We focus on temporal localization, so the sliding windows cover the whole frame. The challenges we face have two causes: presence of background trajectories (including those due to camera motion) and noisy tracklet descriptions. Background and noise features both appear in the BoVW histograms of the frames, potentially overwhelming the relevant information. Thus the classifiers' capacity to discriminate actions from the background is diminished. This phenomenon is aggravated by the GA kernel's atomic comparison function, the L_2 distance. This metric does not place more weight on the action-relevant dimensions of the histograms than on those of the background. Therefore, we propose to use feature selection such that for each action class we choose the visual words that provide the best discriminative power for the GA kernel. Selecting a small subset of features additionally allows to quickly reject non-relevant sequences (those not containing the features) and also speeds up the computation of the kernel. As seen in Seq. 2.2.3, the computational complexity of this kernel is $O(dL^2)$ so keeping a small set of features $S \subset \mathcal{W}$ of cardinality $s << d$ will provide a substantial speedup.

Popular feature selection methods following the filter approach, like mRMR [49], aim to maximize the relevancy of selected features (D, eq. 19) modeled by the mutual information (I) between the selected features (visual word set $S = \{w_i\} \subset \mathcal{W}$) and the class labels ($c \in \{Positive, Background\}$). At the same time, such methods minimize feature redundancy (R, eq. 20). The criterion to maximize is the difference between D and R.

$$D(S, c) = \frac{1}{|S|} \sum_{w_i \in S} I(w_i, c) \tag{19}$$

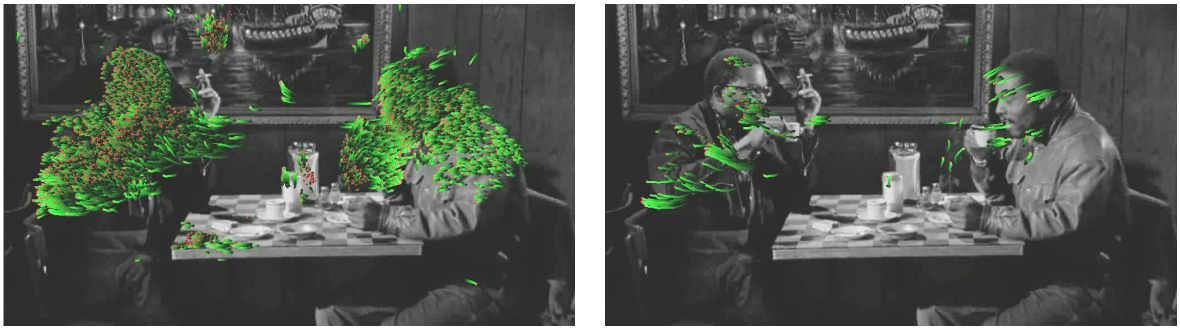

Figure 7: TS-MRMS Feature selection example. Left: all tracked trajectories. Right: trajectories giving selected features (Drinking action, 150 features).

$$R(S) = \frac{1}{|S|^2} \sum_{w_i, w_j \in S} I(w_i, w_j) \tag{20}$$

Another feature selection method following the filter approach, BAHSIC (Backward elimination Hilbert-Schmidt Independence Criterion) [57] finds non-linear correlations between feature values and class labels. The relevance of a feature is measured as the Hilbert-Schimdt norm of the cross-covariance operator between feature maps ϕ, ψ of the original feature values (visual word counts) and labels (c) to two kernel spaces $\mathcal{F}, \mathcal{C}$ (eq. 21). BAHSIC iteratively selects the visual words w_i with the best HSIC score and adds them to the selected feature set.

$$HSIC(\mathcal{F}, \mathcal{C}) = ||\mathbb{E}_{w_i c} \left[(\phi(w_i) - \mu_{w_i}) \otimes (\psi(c) - \mu_c) \right] ||^2_{HS} \tag{21}$$

The high computation cost of the GA kernel and the large initial number of features discourage the use of wrapper approaches for feature selection.

Mutual information and correlation based methods such as mRMR and BAHSIC retain both features (visual words) that are present in the positive examples and absent from the negative ones and features that are present in the negative examples and absent from the positive ones. However, the negative examples for one class include in our case not only examples from the other classes but mostly "background" sequences, i.e. video sequences without any of the actions. Background sequences from a limited training set are not representative of the background in the entire database, which is much more diverse. In this asymmetrical setting, we do not expect mRMR and BAHSIC to perform very well. Furthermore, these methods are designed to select a subset of dimensions in multi-dimensional vectors while in our case we are dealing with time series of vectors. Also, these two methods evaluate each feature individually, while we need to evaluate a feature set as a whole in a time series. We must check if a feature *set* is present or not in a sequence and we do not consider each of the features individually as in eq. 19, 21.

Below, a feature set S will be considered present in a sequence $x = x_1, \ldots, x_L$ if *all* the frames in the sequence contain at least one feature of the set (eq. 22). Also, a feature set S will be considered absent if *at least one* frame contains no feature from the set (eq. 23).

$$Presence(S, x) = \prod_{i=1}^{L} \mathbb{1} \left(\sum_{w \in S} x_{iw} \right) \tag{22}$$

$$\neg Presence(S, x) = \left[1 - \mathbb{1} \left(\prod_{i=1}^{L} \sum_{w \in S} x_{iw} \right) \right] \tag{23}$$

In eq. 22 and 23, $x_{iw} \geq 0$ is the value of word w at time i in the BoVW sequence x and $\mathbb{1}(x) = \{1 \text{ if } x > 0, 0 \text{ otherwise}\}$.

Because of the diversity of content in large video collections and the limited data available for training, such background sequences from the training data are usually *not* representative for other videos. Consequently, features that are present in such sequences and absent from the positive examples will act as noise. Taking this in consideration, a good feature set should appear frequently in the positive training sequences: $P^+(S)$ (eq. 24) should be high.

$$P^+(S) = \frac{1}{|\mathcal{P}|} \sum_{x \in \mathcal{P}} Presence(S, x) \tag{24}$$

with $\mathcal{P}$ denoting the set of positive examples. We take into consideration the fact that we compare sequences of vectors rather than simple vectors. When reducing the visual vocabulary, frames in some sequences will be devoid of any information (all remaining word counts will be 0). Our assumption is that, given a good criterion for selecting the feature set, we will be able to discard all the sequences that contain such frames. To eliminate as many sequences as possible, a good feature set should be absent from the background sequences: $A^-(S)$ (eq.25) should be high.

$$A^-(S) = \frac{1}{|\mathcal{N}|} \sum_{x \in \mathcal{N}} \neg Presence(S, x) \tag{25}$$

with $\mathcal{N}$ denoting the set of negative examples. We aim to find a set of features (visual words) that is (1) maximally present in the positive examples $\mathcal{P}$, and (2) maximally absent from the negative examples $\mathcal{N}$. We call this approach Time Series Maximum Relevancy Maximum Sparsity (TS-MRMS) and propose an incremental greedy algorithm that jointly maximizes the two criteria (Algorithm 1). The objective function to maximize is thus the product of the two frequencies: $P^+(S) \cdot A^-(S)$. Ideally, one would explore all possible feature subsets (the powerset) but, given the number of features available (4,000), this would imply testing all 2^{4000} subsets. We use an incremental method that, at each step, chooses the feature that, when added to the current set, maximizes the objective function. The problem thus becomes tractable. When a validation set is provided we can choose the optimal number of features, i.e. the one giving the best performance on this set. In Fig. 6 this algorithm is represented in block 3.

Algorithm 1 TS-MRMS feature selection algorithm.

Require: A positive set $\mathcal{P}$ and a negative set $\mathcal{N}$ of BoVW timeseries
1: Set S to $\varnothing$, $GroupScore = 0$, $\mathcal{W} = \{\text{all visual words}\}$
2: **while** $Card(S) < s_{max}$ **do**
3: Find $\arg\max_{w \in \mathcal{W}} Score(S \cup \{w\})$ where $Score(\mathcal{G}) = P^+(\mathcal{G}) \cdot A^-(\mathcal{G})$. See eq. 24 and 25
4: Update $S = S \cup \{w\}$, $GroupScore = Score(S \cup \{w\})$, $\mathcal{W} = \mathcal{W} - \{w\}$
5: **end while**
6: **return** S

Note that feature selection criteria introduced by Oikonomopoulos et al. [43] also consider the non-representativity of negative examples, but they use an unordered BoF approach for detection. Our method is adapted to time series and should improve the performance of similarity measures based on time alignment.

When applying feature selection, in some of the windows to be evaluated none of the selected features are present. These windows can thus be discarded even before the first stage (as shown in Fig. 6). Using a feature presence bitmap (1 bit per feature), this test can be done with bit operations to check for presence or absence of the feature set, speeding up the computation with only a slight increase in memory consumption.

Second Stage Classifier. The second stage classifier (Fig. 6, block 5) is applied only to the windows that are considered relevant at the first stage of the cascade and contain the selected feature set. For this second stage we describe windows using the time series of BoVW histograms of all its frames, denoted $X^{(ts)}$. First, sequences not containing the selected features are removed and the remaining ones are reduced (Fig. 6, block 4), keeping only the $Card(S)$ visual words selected by the method in Sec. 2.3.4. Such sequences are compared using the GA kernel and an SVM model is learned giving a decision function f_2. A sequence is relevant to the query if $f_2(X^{(ts)}) \geq 0$. The values of f_2 are used for ranking retrieved sequences.

When localizing specific actions in a large video database, we aim to distinguish a typically rare class (several hundreds of instances) in a very large pool of possibly millions of sequences. While 2-class SVMs can use negative examples to define the decision boundary, in this extremely unbalanced case we cannot assume that the training distribution of negative examples is representative of the one in the testing set. Therefore, we propose to learn a boundary of the domain of the positive examples without using the negative ones. We thus train a One-Class SVM and use only the positive instances as described in Sec. 2.3.2. The SVM learns a hyperplane (the normal w and bias ρ) separating the positives from the "background". The decision function is then:

$$f_2(X^{(ts)}) = \begin{cases} \langle w, \Phi(X^{(ts)}) \rangle - \rho, \text{ if } Presence(S, X^{(ts)}) \\ -\infty \text{ otherwise} \end{cases} \tag{26}$$

Here x_i are the training instances, $\langle \Phi(a), \Phi(b) \rangle$ denotes the scalar product in kernel space and is replaced using the kernel trick by $k_{GA}(a, b)$. Using the dual formulation by introducing Lagrangian multipliers α_i, w can be expressed by a linear combination of support vectors (SVs), giving:

$$\begin{aligned} f_2(X^{(ts)} | Presence(S, X^{(ts)})) &= \left\langle w, \Phi(X^{(ts)}) \right\rangle - \rho \\ &= \left\langle \sum_{i \in SVs} \alpha_i \Phi(x_i), \Phi(X^{(ts)}) \right\rangle - \rho \\ &= \sum_{i \in SVs} \alpha_i k_{GA}\left(x_i, X^{(ts)}\right) - \rho \end{aligned} \tag{27}$$

For training the one-class SVM we need a normalized kernel ($K(x, x) = 1$). Since the GA kernel is not normalized, we apply the following transformation:

$$k'_{GA}(x, y) = \frac{k_{GA}(x, y)}{\frac{1}{2}\left(k_{GA}(x, x) + k_{GA}(y, y)\right)} \tag{28}$$

Post-processing. A sliding window approach for detection can lead to multiple overlapping positive windows. To obtain the final detection sequences, all positive elementary windows (of $L = 30$ frames) whose overlap is above a threshold $\tau_{merge} = 50\%$ are merged by using the union of their bounds. The resulting *detection window* is assigned the sum of scores of the composing elementary windows. In [35] and [18] multiple elementary window lengths of up to 120 frames are employed, which increases computation time. We found that with our method this did not improve results.

2.4 Experimental Results

2.4.1 Data sets

Smoking and Drinking. The Smoking and Drinking data set was introduced by Laptev and Pérez [35] and consists of three videos (resolution 720×576): 2 feature films, "Coffee and Cigarettes" (2002) and "Sea of Love" (1989), and one video consisting solely of drinking sequences. In total it contains 3 hours of video and is split into a training set, a validation set and a testing set (of about 30 minutes).

For the Smoking action there are 78 training, 12 validation and 42 test sequences. For the Drinking action there are 106 training, 16 validation and 38 test sequences respectively.

MSR Action II. The MSR Action II data set introduced by Yuan et al. [71] consists of 1 hour of footage (resolution 320×240) split into 54 videos with cluttered background. It contains three actions selected among those of the KTH data set in Schuldt et al. [55]: Boxing (81 instances), Clapping (51) and Waving (71). Training is done using sequences extracted from KTH (100 videos per action).

MEXaction. The MEXaction data set[7] consists of 117 videos totalling 77 hours, extracted from the digitized archives of the Institut National de l'Audiovisuel. As far as we know, it is the largest collection used for action localization to date. It contains videos produced from 1945 to 2011, digitized from film to 512x384 digital video encoded using MPEG-4 h264 AVC compression.

With respect to the ground truth annotations, the collection contains two actions of interest in the semantic context of a bull fight:

1. BullChargeCape (705 instances). The bull charges the matador who dangles a cape to distract the animal. The actions were annotated to include the bull's charge, the movement of the cape and the feint of the matador.

2. HorseRiding (403 instances). Either during a bull fight or apart, annotations were made of instances of one or several persons riding horses. To restrict the scope of this action class, horse races were however *not* included.

With respect to their content, the videos in this data set can be categorized into: Debate (2), Archive (2), Documentary (14), Soap Opera (1), Interview (15), TV Game (1), TV news (38), Magazine (24), Film press (9), Biographic (2), Report (5) and TV show (4).

Annotations include attributes giving information about the difficulty of the detection as perceived by the annotator: it is harder to detect actions that occupy little of the image plane or actions that are occluded.

The data set contains 6.5 M frames in total, so sampling one $L = 30$ frame sequence every 5 frames we obtain 1.3 M windows. We generate 10 splits for the training and validation sets. Thus, on average, we use 85 BullChargeCape and 50 HorseRiding annotations and 80,000 background frames for training. For hyper-parameter (s, C, σ, τ_1) selection we obtain, on average, 70 BullChargeCape and 30 HorseRiding annotations and 80,000 background frames. This leaves 570 annotations for BullChargeCape and 344 for HorseRiding, with 6.3 M background frames in the testing part of the data set. The training and validation annotations are chosen among those that do not present a high difficulty of detection (i.e. occupy most of the image plane, are not occluded).

2.4.2 Evaluation metrics

Following the recent literature, action localization is evaluated as a retrieval problem: we consider all detection windows with positive scores as results and we sort the windows by their score. This allows to obtain precision/recall curves and to compute the Average Precision (AP) in order to characterize the detection performance.

To quantify the speed-up achieved by the cascaded approach we measure the testing time of the two stages individually and in cascade, as well as the ratio of sequences that need to be evaluated at the second stage (the *coverage*, Sec. 2.3.3). We study the trade-off between the False Negative Rate and the *coverage* on the validation set and we vary the number of features that are selected to find the optimal performance.

First, a definition of positive and negative detections is needed. For the Smoking and Drinking and MSR Action II data sets, actions are sparsely spaced in the timeline of the videos. Existing research [35] thus considers a positive detection A to be a true positive if the Jaccard coefficient $\mathcal{J}(A, B) = |A \cap B|/|A \cup B|$ between it and a positive ground truth annotation B is greater than

[7] http://mexculture.cnam.fr/xwiki/bin/view/Main/Datasets

Table 2: Key figures for the first stage on MSR Action II and Smoking and Drinking

	Windows with action	Coverage	FNR_1
Smoking and Drinking (1 train / validation split)			
Drinking	5%	35%	5%
Smoking	7%	70%	0%
MSR Action II (1 train / validation split)			
Clapping	9%	48%	0%
Waving	12%	35%	0%
Boxing	8%	46%	0%

Table 3: Key figures for the first stage on MEXaction

	Windows with action	Coverage	FNR_1
MEXaction (10 train / validation splits)			
BullChargeCape	0.24%	$8.1 \pm 4.0\%$	$23.8 \pm 4.5\%$
HorseRiding	0.40%	$9.1 \pm 1.7\%$	$27.8 \pm 4.6\%$

20%. These data sets were initially created for spatio-temporal localization, so some actions are overlapping in time, while being separated in space. We aim to localize actions only in time, so the rule $\mathcal{J}(A, B) \geq 20\%$ is applied for temporal overlap. A positive detection can thus match several ground truth windows (separated in space) if the temporal overlap satisfies this rule, as in [18],[32],[44]. If several positive detections overlap a ground truth window by more than 20%, the one with the highest overlap is considered a true positive while the others are considered false positives.

For the MEXaction data set the actions are rare but occur in clusters in which they are separated from each other by few frames. Sometimes actions appear successively in a chain. Thus, a detection window is likely to cover several ground truth annotations. We then adopt a more relaxed evaluation criterion. Ground truth annotations B_i are marked as detected if they are at least 50% inside a detection window A, $|A \cap B_i|/|B_i| > 0.5$, and cover at least 20% of the time span of the detection window: $\sum_i |A \cap B_i|/|A| > 0.2$.

2.4.3 Results and discussion

We begin by evaluating the filtering performance of the first stage classifier on the different data sets. Smoking and Drinking is small and has few action instances, so we select the first stage decision threshold τ_1 to minimize solely the false negative rate (FNR_1). On MSR Action II, no validation split is provided so τ_1 was set to the mean first stage decision value. For these data sets, the impact of the first stage is summarized in Table 2. The average coverage (ratio of sequences sent to the second stage) is of 47%. But these data sets are "dense", i.e. the average ratio of windows containing the actions is of 8%. Table 3 gives the results for our novel data set, MEXaction. Here, the average percentage of windows containing the actions is of 0.32%, i.e. one order of magnitude lower than for the first two data sets. MEXaction is much less "dense", while containing many more instances of each action. We thus jointly minimize the coverage and FNR_1, obtaining a much lower average coverage of only 8.6%. We nevertheless lose part of the action instances, as shown in column FNR_1 of Table 3. For a data set with very many action instances this trade-off may be acceptable.

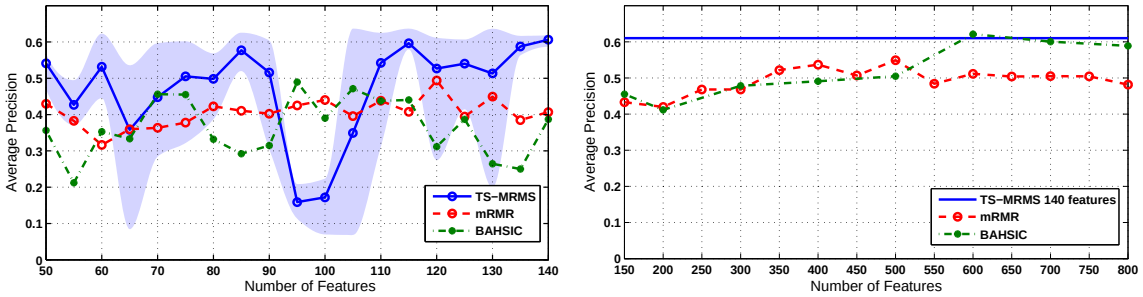

Figure 8: AP of GA classifier: comparison of TS-MRMS, mRMR and BAHSIC feature selection algorithms. Left: same number of features. Right: more features for mRMR and BAHSIC (150-800) compared to 140 for TS-MRMS

We now analyze the behavior of the proposed TS-MRMS feature selection method and we compare it to mRMR and BAHSIC (in the binary classification setting described in [57]) and to using all the features. The evaluation is done on the Smoking and Drinking data set and we look at the impact of varying the number of features on the detection performance for the Drinking action. For this experiment we tested the GA classifier in a 2-class SVM setting, varying the negative instances used for learning.

In Fig. 8 we illustrate the average AP over 5 trials when varying the number of features for the three methods. We also show the interval between minimum and maximum values (shaded) for TS-MRMS. For 50-140 features (Fig. 8, left) the TS-MRMS method has consistently better performance than mRMR and BAHSIC, with good stability over trials. An exception is at 100 features where the performance of TS-MRMS shows an anomalous drop. We show the behavior of mRMR and BAHSIC when further increasing the number of features in Fig. 8, right. For mRMR only a small gain in performance is achieved, AP reaches a maximum of $54.9\% \pm 4.2\%$ at 500 features while the best performance with 140 TS-MRMS features is $60 \pm 1\%$. The same AP is obtained by BAHSIC with 600 features. As a baseline, without feature selection, we found that using all 4,000 features gives an AP of $46 \pm 4\%$.

On the Smoking and Drinking and MSR Action II data sets, feature selection strongly reduces the number of features, from 4,000 to 30-150 (Table 4). On MEXaction, as shown in Table 5, the number of features for the GA classifier is even lower: since the database is very large, the tracklet descriptors are extremely varied and the ones relevant to the actions of interest are rare. This significantly accelerates the second stage of the cascade since the computation of the GA kernel has a complexity of $O(sL^2)$ (L is the window length, s the number of selected features). With regards to the filtering performance, the rows "Windows with features" in Tables 4 and 5 show the overall ratio of windows that contain the selected features and must be evaluated by the SVM classifiers.

Table 4: Feature selection pruning results on MSR2 and Smoking and Drinking

(1 train / validation split)	Drinking	Smoking	Waving	Clapping	Boxing
No. 2^{nd} stage features	150	150	50	30	30
Windows with features	93%	89%	83%	37%	72%

Next, we compare the detection performance of the cascade to the first stage classifier alone and to the GA kernel classifier alone. Table 6 summarizes the AP results for the three datasets. The Precision-Recall curves are illustrated in Fig. 9, Fig. 10 and Fig. 11 respectively. The performance of the cascade is systematically better than the performance of the GA kernel classifier and, with the exception of the Clapping action, better than the performance of the first stage classifier (the result

Table 5: Feature selection pruning results on MEXaction

(10 train / validation splits)	BullChargeCape	HorseRiding
No. 2^{nd} stage features	10-40	12-40
Windows with features	$33.0 \pm 4.7\%$	$33.9 \pm 5.0\%$

of the GA kernel classifier on Clapping could probably be improved by choosing longer detection windows). The average improvement is of 10.5% with respect to the first stage alone and of 7.3% with respect to the GA kernel alone. Thus, beside reducing cost, the cascade also improves the detection quality, which shows the complementarity of the two classifiers.

Table 6: Detection AP of the cascade and individual stages

Action	Clapping	Waving	Boxing	Smoking	Drinking	BullCharge	HorseRiding
1^{st}	**43.1%**	26.2%	25.3%	62.1%	21.6%	31.7%	18.2%
2^{nd}	32.3%	40.0%	27.4%	64.4%	38.3%	30.0%	18.1%
Cascade	39.7%	**55.0%**	**39.6%**	**65.5%**	**45.1%**	**37.7%**	**19.5%**

We analyze the performance and cost differences when using one-class SVMs for a large scale data set. Table 7 shows that we obtain a large gain in AP with fewer support vectors on MEXaction. We believe this is due to the difference in ratios of positives to negatives in the training set relative to the testing set. For Smoking and Drinking the training and testing sets come from the same video and are approximately of equal size. For MEXaction the testing set is 80 times larger and only 0.2-0.4% of windows are positive (Table 2). Moreover, the MEXaction testing set contains videos that are not used for training, thus negative examples from the training set are not necessarily relevant, making one-class SVMs a better choice.

Table 7: Comparison for the GA classifier: 2-class SVM vs. 1-class SVM, on Smoking and Drinking and on MEXaction

Method		Drinking	Smoking	Bull Charge Cape	Horse Riding
2-class SVM	AP	**64.4%**	**38.3%**	$9.7 \pm 2.3\%$	4%
	nSV	1961	1097	2200-2800	600
1-Class SVM	AP	52.9%	27.6%	**28.3±2.6%**	**15.5±2.8%**
	nSV	114	274	20-400	60-290

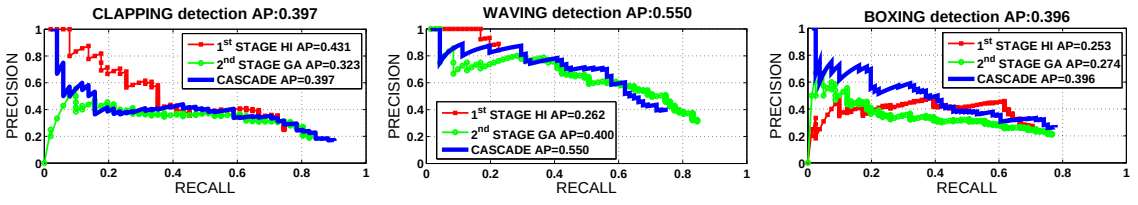

Figure 9: Precision-recall curves on MSR Action II

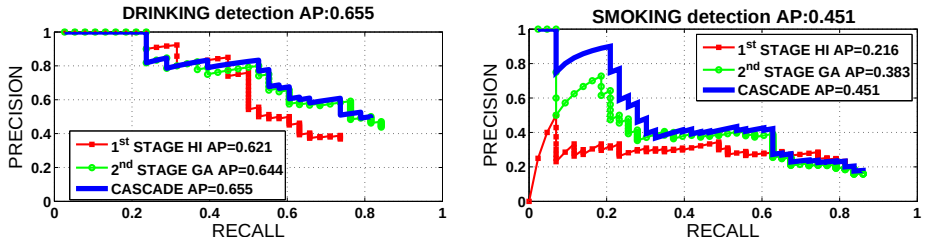

Figure 10: Precision-recall curves on Smoking and Drinking.

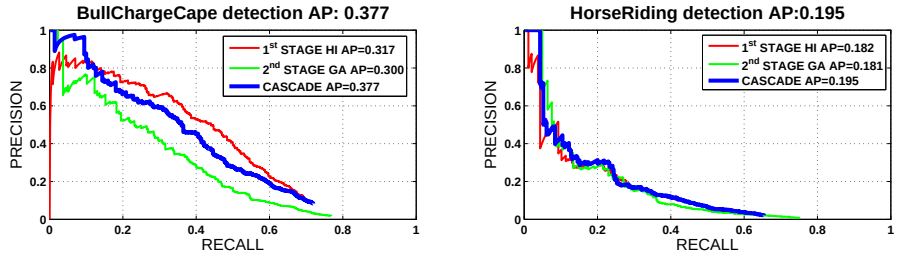

Figure 11: Precision-recall curves on MEXaction.

The action localization performance of the cascade on MSR Action II is shown in Table 8, together with the current state of the art results. Previous work on MSR Action II in [8],[68],[69] focuses on spatio-temporal localization, so a direct comparison cannot be performed. While the cascade was not designed for fine spatio-temporal localization, our experiments show that it has good temporal localization results on MSR Action II where actions cover only a small part of the video frame.

Finally, we compare the action localization performance of our method to the state of the art results on Smoking and Drinking, as well as to a state of the art method (Fisher Vectors, FV) on MEXaction (Table 10). We used the FV implementation from the VLFeat library[8], computing the vectors on the tracklet features in $L = 30$ frame windows. As suggested by Perronnin et al. [50] ("improved" Fisher Vectors), we applied both the power and L2 normalizations to the vectors. On the Smoking and Drinking data set (Table 9) our results are superior to the state of the art on the Drinking class. Most importantly, our method shows a good improvement over the baseline (FV) on both classes of the MEXaction data set while also being better adapted to the large size of the database. Sample frames from detections are shown in Fig. 12 and 15 for MEXaction and in Fig. 13 and 14 for Smoking and Drinking. We show some of the highest ranked true positives (TP), as well as the first and second highest ranked false positives (FP) with their ranks in the result list.

Table 8: Performance on MSR Action II

Method	Metric	Clapping	Waving	Boxing
Cao et al. [8]	Spatio-Temporal	13.1%	36.7%	17.5%
Yu et al. [68]	Spatio-Temporal	23.9%	43.0%	30.3%
Yu et al. [69]	Spatio-Temporal	36.1%	54.1%	31.7%
Cascade AP	Temporal	39.7%	55.0%	39.6%
Cascade search time		78 s average		

[8] http://www.vlfeat.org/

Table 9: AP comparison on Smoking and Drinking

Method	Drinking	Smoking
Laptev and Pérez [35]	49%	–
Gaidon et al. [17]	57%	31%
Klaser et al. [32]	59%	24%
Oneata et al. [44]	64%	**50%**
Cascade (with 2-class SVM)	**65.5%**	45.1%
Cascade search time	178 s	240 s

Table 10: Performance comparison on the MEXaction data set

Method		BullChargeCape	HorseRiding
GA classifier (1C)	AP	$28.3 \pm 2.6\%$	$15.5 \pm 2.8\%$
	Time	2084s	2002s
	DB	∼250 MB	
Cascade	AP	**33.5± 3.5%**	**17.6±1.9%**
	Time	265s (159+106)	247s (150+97)
	DB	3.5 GB	
Fisher Vectors	AP	$31.1 \pm 5.9\%$	$14.0 \pm 1.4\%$
	Time	43 s	
	DB	500 GB (80 GB w. PCA)	

In Tables 8, 9 and 10 the computation time of our method is presented for the three data sets. We used GPU implementations of the GA and HI kernels running on an NVIDIA Tesla C2070 card. The evaluation server had 2 Xeon CPU and 24 GB of RAM. Note that by "computation time" we only refer to the time required to compute the scores of all elementary windows provided the test sequences are loaded into RAM. For MSR Action II and Smoking and Drinking we employed 2-class SVMs while for MEXaction we used one-class SVMs. The speed-up results obtained by our method are summarized in Table 10: with a cascade having good pruning power (Table 3), a low number of features (Table 5) and a low number of support vectors in the second stage (Table 7), the computation time for MEXaction is only slightly larger than the one for Smoking and Drinking, even though the data set is 150 times larger. The total query time with the cascade is expressed in Table 10, line "Time", as a sum between the time spent in the first and second stages; the first stage takes about 60% of the query time.

The size of the descriptor database (rows labeled DB in Table 10) is an important factor in action retrieval from large collections. For our method, using 4,000 visual word histograms, the 6.5 M frame descriptors need 10 GB of storage for the aggregated sequences and 50 GB for the time series representation. Of the video time series only small parts have to be loaded in memory: the sequences containing the selected feature set and, for these sequences, only the values of the features in the set. Thus for BullChargeCape, when the feature set contains between 10 and 40 features we need to load about 200MB of data to determine which sequences to test. Using this information, between 8 MB and 32 MB of data need to be loaded for the second stage and 3.3 GB for the first one. The FV method tested requires 101,376 dimensions per sequence (we used K=128 Gaussians and the tracklet descriptor has 396 dimensions) needing 500 GB of storage. In [44] the number of dimensions is reduced to 16,000 by applying PCA to the tracklet descriptors. For exhaustive search on MEXaction set this would lead to a database size of 80 GB that would need to be fully loaded into RAM. When considering a disk read speed of 100 MB/s, we can see that the time to load the FV database (about 13 minutes) dominates the retrieval time, while using our method only 30 s are necessary.

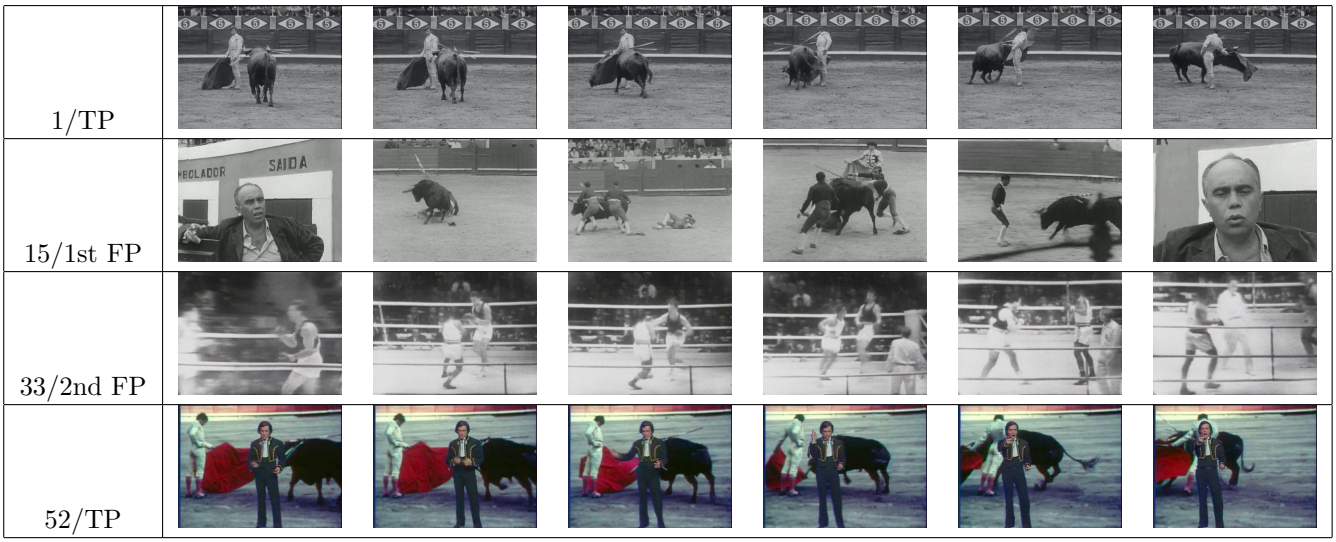

Figure 12: Sample results for the BullChargeCape action of the MEXaction data set.

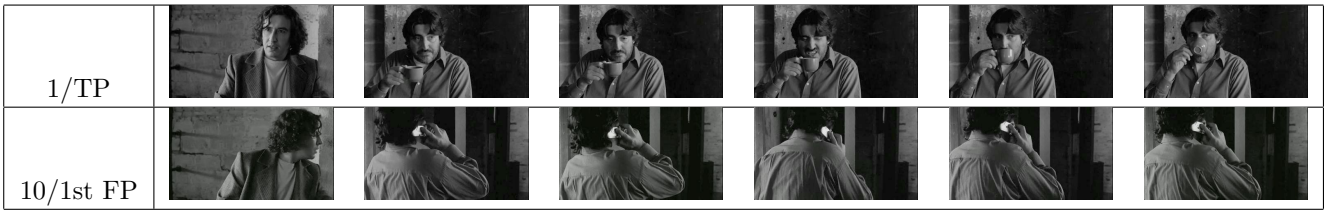

Figure 13: Sample results for the Drinking action of the Smoking and Drinking data set.

2.5 Conclusion and Perspectives

We presented a cascaded system for action localization that shows good performance and fast retrieval of actions in large data sets. A substantial gain was obtained by using an adequate feature selection method for the time alignment of sequences with the GA kernel. This also lead to a large reduction in the amount of data that has to be loaded from the disk. The proposed method achieves better performance than the state of the art while having lower memory requirements.

We introduced the MEXaction data set that is a very relevant benchmark for large scale action localization due to its size and content variety. In this context, negative training examples may not capture the full diversity of non-action sequences in the whole data set and can reduce the discriminative power of classifiers. We showed that one-class SVMs can address this problem and give good retrieval accuracy with few support vectors, allowing fast detection. However, a learning method that better exploits the negative examples should be further explored for large scale action localization using GA-kernel classifiers. Finally, to perform real-time queries for realistic video collections, indexing methods should be further considered in order to make the search of relevant windows sub-linear in the size of the database.

2.6 Sublinear Retrieval

Large video databases contain many action classes of potential interest. For example, actions are very relevant content items in historical and cultural videos; a historian, a researcher or some other user should be able to define action classes and search for occurrences of such actions in a large cultural database. In a video-surveillance application, an investigator may also have to define new specific action classes and search for their occurrences in a potentially high volume of video records.

Figure 14: Sample results for the Smoking action of the Smoking and Drinking data set.

Figure 15: Sample results for the HorseRiding action of the MEXaction data set.

In such scenarios, it can be prohibitively time consuming or excessively expensive to perform a new exhaustive scan of the entire database every time a new class detector is built. It is then necessary to devise methods supporting the *scalable* application of a detector to the data, i.e. methods that are sublinear in the size of the database. Such methods should avoid an exhaustive scan and only apply the detector to a (hopefully) small part of the data, where the detector is relatively likely to provide a positive answer. While the scalability of query-by-example was thoroughly considered in the literature, there is comparatively little work on the scalability of what we shall call *query-by-detector*. In this work we address this problem by proposing a method that supports sublinear retrieval of complex human actions.

Action detection and localization was addressed in the recent literature (see e.g. [14, 17, 44, 45]), but not on large scale datasets. Actions are usually represented by the statistical distribution of local features that describe shape and motion in video patches. In [14, 17], Bags of Visual Words (BoVW) histograms are used to model the statistics of features in video segments, while [44, 45] employ instead Fisher Vectors. Both approaches involve high dimensional vector descriptions. Support vector machine (SVM) classifiers are learned from the annotated examples and detection is performed by applying the SVM to a window that slides over the entire video database. Thus, all the windows in the database are evaluated by the detector and the positive detections are eventually ranked to produce the result list.

There exist more efficient alternatives for applying a detector to a database. A method for approximating a linear SVM decision function using locality-sensitive hashing (LSH [11]) is introduced in [37]. The normal vector to the SVM hyperplane is used as a query and its hash is obtained. The approximation to the decision function employs the Hamming distance between the query hash and the hashes of the non-empty buckets. The complexity is linear in the size of the database but the approximation allows to significantly accelerate the retrieval of the data points classified as positive by the detector.

To best deal with large databases, exhaustive detection (i.e. sliding the detection window over the entire video database) should nevertheless be avoided. Several proposals focus on sublinear methods that aim to find the data points whose image in feature space is close to the normal vector to the SVM hyperplane. Since for most kernels employed $\mathcal{K}(x, x) = \text{constant}$, these points maximize the SVM decision function. The KDX index structure in [47] defines rings around the normal vector and indexes them according to the angle to the central vector. A second level of the index is used within each ring. A sublinear exact search solution for such SVM-based queries was proposed in [31] and claimed to improve over KDX. The data is clustered in feature space and, for each cluster, rings are built with the kernel space neighbors of the cluster prototypes, by order of their distance. Querying this index structure with the normal vector requires testing all prototypes and then the corresponding cluster rings close to the query in order to accumulate the results. However, as we shall see later, the relevant data to be retrieved is not necessarily close to the normal vector in feature space. We can also mention here the method in [30] for exact sublinear retrieval with hyperplane queries, shown to be efficient but in rather low-dimensional spaces.

Alternatively, SVM-based active learning using an ambiguousness criterion requires retrieval of unlabeled data that is close to the decision boundary (which is a hyperplane in feature space). If the amount of unlabeled data is large, sublinear retrieval methods are needed. Several such solutions, based on LSH, are proposed in [26, 38, 60]. They rely on the fact that data points that are close to the decision hyperplane have a low inner product with the normal vector to the hyperplane. Such approaches can be interesting if the target class is defined online, e.g. with relevance feedback, but this is not the scenario we consider here.

In Section 2.7 we first show how to adapt the approach in [37] to create a linear time "exhaustive approximate" (EA) search method for queries that are nonlinear SVM detectors. We then introduce a novel approximate sublinear method for answering such queries. In Section 2.8 we provide the experimental validation of our sublinear method and compare it to both exhaustive approximate and exhaustive exact (EE) search.

2.7 Proposed approach

We start by briefly presenting the general action detection and localization method we employ, then we focus on the scalability issue where our contributions are.

2.7.1 Action localization method

Video description. We employ the method in [62] to detect and describe salient video patches. We quantize these descriptors into a visual dictionary of $d = 4000$ words and compute an L1 normalized BoVW histogram for each frame. The videos are over-segmented into a series of time windows of $L = 30$ frames, shifted by 5 frames. For each window, the frame histograms are averaged to produce its final description.

Action detection in sliding windows. For detection, an SVM classifier is trained for each action class on annotated examples. The SVM decision function used to score results is

$$f(v) = \sum_{j=0}^{p} \beta_j \mathcal{K}(y_j, v) + b_q \tag{29}$$

Since each window is described by a BoVW histogram, we use the histogram intersection (HI) kernel:

$$\mathcal{K}(x, y) = \sum_{i=1}^{d} \min\{x_i, y_i\} \tag{30}$$

Note that $\langle \Phi(x), \Phi(x) \rangle = \mathcal{K}(x, x) = 1$. All the windows in the database are evaluated by f and the positive detections are retained for further processing. We use this "exhaustive exact search" (EE search) as a baseline. Its complexity is linear in the size of the database.

Post-processing. Multiple overlapping windows can have positive detection scores. To obtain the final detection boundaries, all positive windows whose overlap is above a threshold $\tau_{merge} = 50\%$ are merged by using the union of their bounds. The resulting *detection window* A is assigned the sum of scores of the composing windows: $S(A) = \sum_i f(v_i)$.

2.7.2 Exhaustive approximate search

We first adapt the method in [37] to kernel space hashing using the Random Maximum Margin hashing (RMMH) functions proposed in [28]. With RMMH, LSH functions are hyperplanes in the feature space. To construct each hyperplane, M data points are chosen at random and randomly labeled as positive or negative examples. An SVM is learned from these examples and its decision function is one atomic hash function used to build the index:

$$h(v) = \text{sgn}\left(\sum_{i=0}^{m} \alpha_i^* \mathcal{K}(x_i^*, v) + b \right) \tag{31}$$

where x_i^* are the m support vectors of one hash SVM, α_i^* their Lagrangian multipliers and $\mathcal{K}(x, y)$ is the employed kernel. A hash table is built from D such atomic functions obtained on independent random samples of M points. To retrieve nearest neighbors, the query q (in input space) is hashed and its hash $H(q)$ is computed as the concatenation of the atomic hash values: $\{h_1(x), ..., h_D(x)\}$. The probability that the hash value of data v is equal to that of the query q is proportional to the inner product between $\Phi(v)$ and $\Phi(q)$ [20]:

$$Pr\left[h(q) = h(v)\right] = 1 - \frac{1}{\pi} \cos^{-1}\left(\frac{\Phi(q) \cdot \Phi(v)}{||\Phi(q)||\,||\Phi(v)||} \right) \tag{32}$$

We build D RMMH functions with the HI-kernel and compute the hash of each data v_i, $i \in \{1..N\}$, obtaining binary vectors $H(v_i)$. However, in our case the query corresponds to the SVM detector in

Eq. (29) and is represented by the normal vector q of the learned hyperplane in *feature* space. This vector is a linear combination of the feature space images of the learning examples, $q = \sum_{j=1}^{p} \beta_j \Phi(y_j)$, so its discrete hash value $h(q)$ can be computed according to:

$$\begin{aligned} g(q) &= \sum_{i=0}^{m} \alpha_i^* \left\langle \Phi(x_i^*), \sum_{j=1}^{p} \beta_j \Phi(y_j) \right\rangle + b \\ &= \sum_{i=0}^{m} \alpha_i^* \sum_{j=1}^{p} \langle \Phi(x_i^*), \beta_j \Phi(y_j) \rangle + b \\ &= \sum_{i=0}^{m} \alpha_i^* \sum_{j=1}^{p} \beta_j \mathcal{K}(x_i^*, y_j) + b \\ &= \vec{\alpha^*} K \vec{\beta}^T + b \\ h(q) &= \operatorname{sgn}(g(q)) = \operatorname{sgn}\left(\vec{\alpha^*} K \vec{\beta}^T + b\right) \end{aligned} \tag{33}$$

where $K_{ij} = \mathcal{K}(x_i^*, y_j)$, x_i^* ($i \in \{1..m\}$) are the support vectors of the RMMH function, y_j ($j \in \{1..p\}$) are the support vectors of the action detector, $\vec{\alpha^*} = (\alpha_1^*, ..., \alpha_m^*)$ and $\vec{\beta} = (\beta_1, ..., \beta_p)$.

We take from [37] the approximate decision value of the detector, $\hat{f}_q(v)$, as a function of the Hamming distance between the hashes of the query and of the data point, $d_H(H(q), H(v))$:

$$\hat{f}_q(v) = \cos\left(\pi \frac{d_H(H(q), H(v))}{D}\right) ||q|| + b_q \tag{34}$$

We do not need to compute $\hat{f}_q(v)$ for any v, we just sort the buckets in ascending order of their Hamming distance to the query. All data points in the top buckets are retrieved and evaluated by the decision function in Eq. (29). As many buckets are returned as necessary to reach the desired recall. Note that $\operatorname{sgn}(\hat{f}_q(v))$ does not give accurate positive detections, the estimator can have a large probability of false positives.

2.7.3 Scalable retrieval

The method introduced in Section 2.7.2 can be efficient enough for medium-size video databases but requires the evaluation of the Hamming distance between the hash of the query and every bucket. For very large databases, the number of buckets should increase linearly with the size of the database. So the complexity of this method is linear in the size of the database. To scale to very large databases it is then necessary to devise sublinear methods.

If the feature-space images of the data points for which f (the decision function of the SVM detector) takes positive values were in the neighborhood of the normal vector to the SVM hyperplane, then a potentially good solution would be to use a method like Multi-probe LSH [39]. Multi-probe LSH proposes to search those buckets whose spatial boundaries are close to $g(q)$. Assuming the hash functions are independent, each $h(q)$ value can be flipped to produce hashes that have lower probability than the query's hash. In [39] a probing algorithm that searches buckets in decreasing order of their probability is given. Multi-probe LSH allows to reduce the number of different hash tables required to reach a given level of recall.

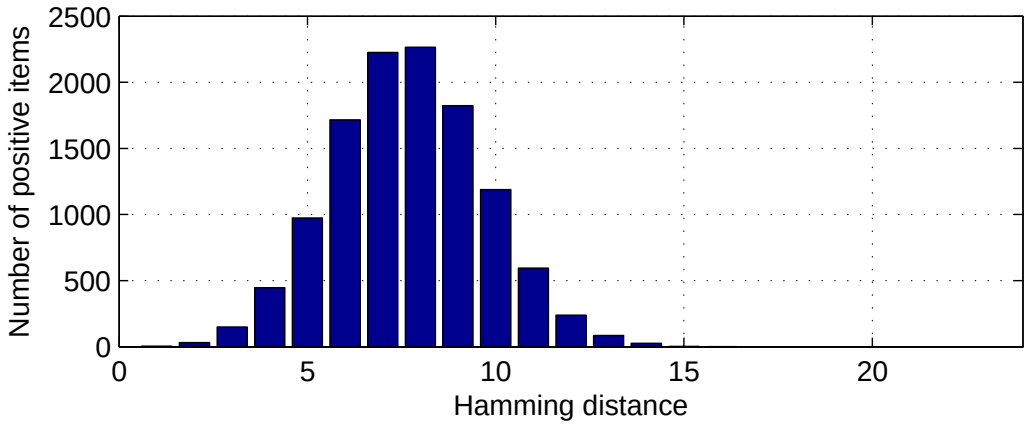

Figure 16: Positive results are far from query's hash bucket

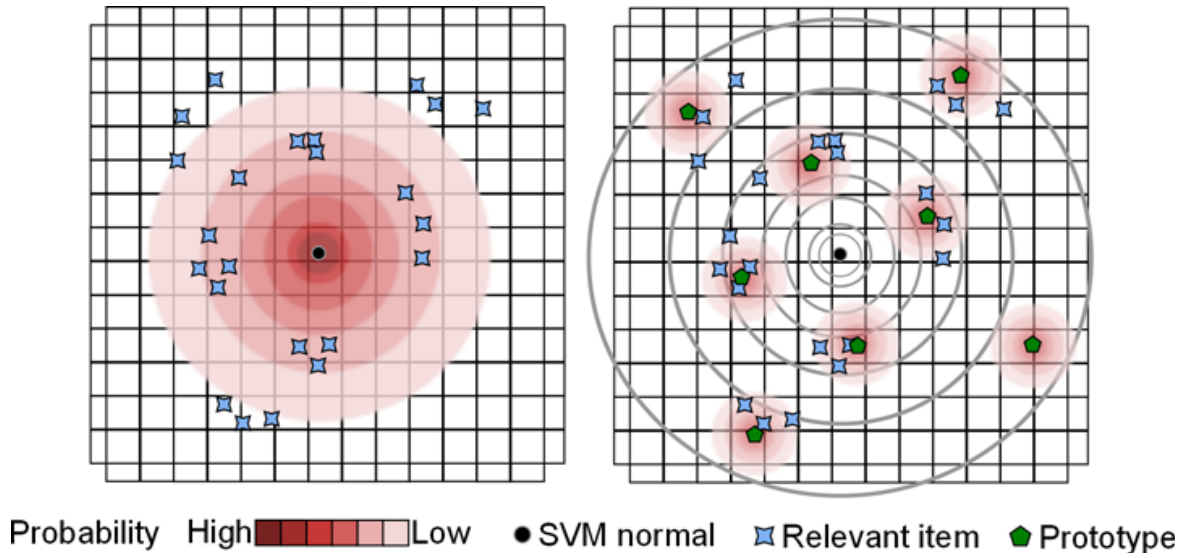

Figure 17: Left: probing from the query requires traversing a large number of buckets to reach positive data points. Right: probing from prototypes should bring us closer to the positives

We evaluated the distribution of Hamming distances between the hashes of the data points for which f takes positive values and the hash of the query. Figure 16 shows that the maximum of this distribution is far from $d_H = 0$. To arrive at a Hamming distance $s = d_H(H(q), H(v))$, for one hash table, Multi-probe LSH should explore $\sum_{n=1}^{s} \binom{D}{n}$ hashes. The assumption made by Multi-probe LSH that the probability of finding relevant results is an isotropic normal distribution centered on the query is clearly wrong; see the illustration on the left side of Figure 17.

Since the query itself is not a good starting point, we propose to use instead the prototypes of the positive training examples as starting points. We use kernel K-means clustering to obtain K clusters on the positive training examples and for each cluster k we obtain the most central element, the cluster prototype p^k:

$$p^k = \text{argmax}_j \sum_{i=1}^{n^k} \mathcal{K}(y_j^k, y_i^k) \tag{35}$$

Our assumption is that cluster prototypes are more representative of the positive data points than the query (the normal vector to the SVM hyperplane). So, by starting to sample from the buckets of the prototypes should allow to reduce the total number of probes needed to reach a required level of recall. Finally, we apply the multi-probe algorithm of [39] for each of the prototypes.

The method we propose is illustrated in the right side of Figure 17. This picture shows the case of using two hash functions that give a range of integer values. In our case we have D hash functions that give only binary values, but the idea is the same. Note that the advantage of using cluster prototypes over cluster centers is in computation time. For a cluster center we would need to use Eq. (33) while for the prototypes we can directly use Eq. (31).

Cost reduction with query expansion. To increase recall we can increase the number of clusters and the probes around each cluster prototype. However, further probing is computationally expensive, while data points that are farther from the prototypes have lower probability of being positive. A better informed way to direct the search for candidates would be helpful. In the context of action localization we can use the fact that temporal neighbors of windows (data points) detected as positive have a high probability of being positive themselves. Also, since we have the $H(v)$ values of the temporal neighbors, we can apply a simple filter by using Eq. (34). This filtering is fast and has a very low false negative rate, so it eliminates only a negligible number of good candidates. The approximate test is only applied to the temporal neighbors of windows already detected as positive.

For details, a listing of the algorithm of our sublinear method can be found on our website[9].

2.8 Experimental evaluation

Dataset. We evaluate the effectiveness and efficiency of our action localization method on the Corrida dataset[10] of 77 hours of video. With the video representation described in Section 2.7.1, the database contains 1.3M windows, each represented by a 4000-dimensional BoVW histogram. The annotations for two actions were employed: (1) BullChargeCape—in the context of a bull fight, the bull charges the torero who dangles a cape to distract the animal, and (2) HorseRiding—one or several persons riding horses. The dataset is split into two parts. For training and parameter validation, there are 2 hours of video containing 85 examples of BullChargeCape and 50 of HorseRiding. For testing, there are 75 hours of video in which we attempt to identify 570 instances of BullChargeCape and 344 of HorseRiding.

Metrics. Following current practice, we evaluate action localization like a retrieval problem: detection windows A having positive scores S(A) are sorted by decreasing scores and a precision - recall curve is obtained. Average Precision (AP) is then computed for each class. A result window is positive if it overlaps the ground truth annotations. We aim to localize actions only in time, so only temporal overlap is measured. Ground truth annotations B_i are marked as detected if they are at least 50% inside a detection window A, $|A \cap B_i|/|B_i| > 0.5$, and successive annotations cover at least 20% of the time span of the detection window: $\sum_i |A \cap B_i|/|A| > 0.2$.

[9] http://cedric.cnam.fr/∼stoiana/supp.pdf
[10] http://mexculture.cnam.fr/xwiki/bin/view/Main/Datasets

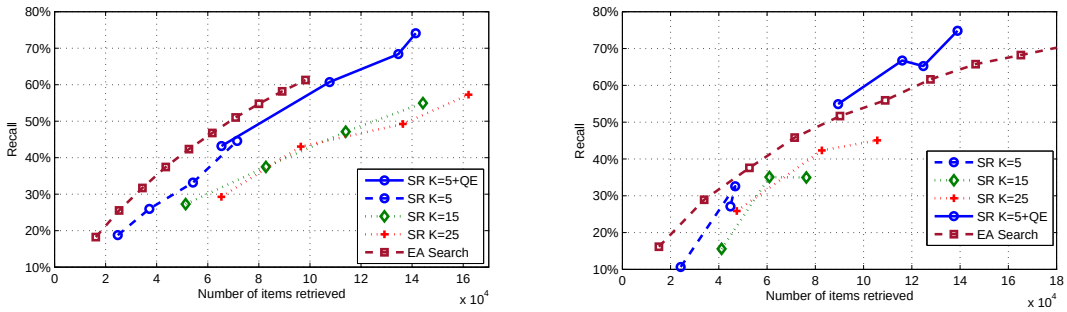

Figure 18: Recall comparison for BullChargeCape (left) and HorseRiding (right). Lines correspond to variable numbers of probes at fixed K

For the experiments we ran, the number of hash tables was $L = 16$ and the number of hash functions $D = 24$ and $M = 32$.

We first measure the recall while varying the number of prototypes $K = \{5, 15, 25\}$ and the number of probes $P = \{5, 10, 20, 30\}$ (Figure 18). We compare the scalable retrieval (SR below) method in Section 2.7.3, with or without query expansion (QE), to the exhaustive approximate (EA below) search in Section 2.7.2 and take the exact exhaustive (EE below) search as a reference. Thus, for a window retrieved by SR or by EA search, we check its $f(v)$ value to see if it was truly a positive result. Recall is then measured as the number of positive items retrieved divided by the total number of items marked positive by EE Search.

For the BullChargeCape class, the results show that SR obtains the recall of EA search by testing only a small additional number of data points. Moreover, we see that by using QE and $K = 5$ clusters we obtain better results than by using more clusters. For HorseRiding, by using QE (label 'K=5+QE') we have higher recall with fewer data points retrieved than by EA search. For these results we did 2 query expansion iterations.

We now check the Average Precision for localization of SR (Figure 19). For $K = 5, P = 30$ with QE (label 'K=5+QE') for BullChargeCape SR achieves almost the same AP as the EE method while examining 11% of the database. Note that the percentage of data points considered positive according to EE is 4.3% of the database. With $K = 25, P = 30$ we obtain the same AP as EE search. SR examines 3% more data points than EA search to achieve the same AP.

For HorseRiding, neither method reaches the AP of the EE search but SR achieves 10% AP (average over 5 runs) by examining 11% of the database. For this class, the percentage of data points considered positive according to EE is 3.3% of the database. Again, SR examines 3% more data points than EA search to achieve the same AP. The major difficulty for this class is the quality of the detector (EE search): it only provides an AP of 18.3%. Even though we obtain high recall with respect to the detector (75%) with both SR and EA search (Figure 18), the post-processing window fusion method appears to be too sensitive to the absence of the remaining 25% of the positive video sequences.

We give a theoretical analysis of the complexity of our method. To run the SR method with query expansion we first need to hash the query (SVM normal vector). This takes $L \times D \times N_{SV}$ kernel computations. Next, to hash the prototypes of the clusters, we need to run $K \times L \times D \times M$ kernel computations. The complexity is thus dependent on the number of training vectors of the query SVM. For EA search the complexity is linear in the size of the database, even though only Hamming distances are to be computed. Thus, we expect SR to scale better to very large databases.

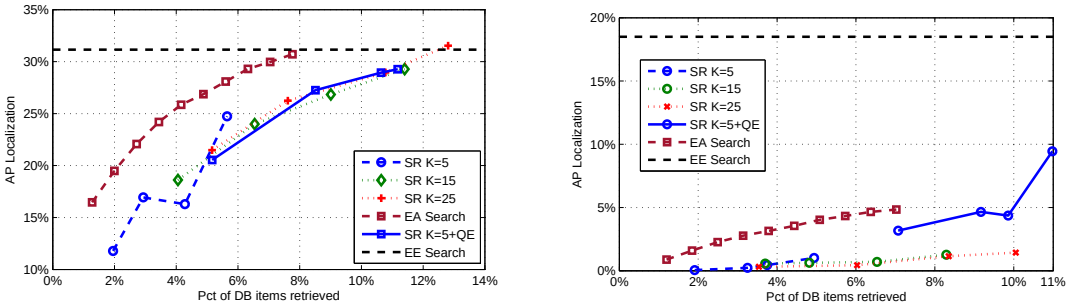

Figure 19: Localization Average Precision: BullChargeCape (left) and HorseRiding (right)

2.9 Conclusion

We presented a method that allows to find in a potentially large database most of the instances of a complex class without having to check more than a fraction of the data. The class is defined here by an SVM detector obtained on training examples. We showed that this method can approach the effectiveness of exact exhaustive search while being much more efficient since it only examines a fraction of the data. The method is not dependent on kernel type and parameters, nor on database size. For databases of medium size, we have also shown that an approximate exhaustive search method can be faster. To improve upon this work, we plan to explore better ways to sample from the distribution of potentially relevant hash buckets.

References

[1] Jurandy Almeida, Neucimar J. Leite, and Ricardo da S. Torres. Online video summarization in compressed domain. *Journal of Visual Communication and Image Representation*, 24:729–738, 2013.

[2] Ilaria Bartolini, Marco Patella, and Guido Stromei. The Windsurf Library for the Efficient Retrieval of Multimedia Hierarchical Data. In *Proceedings of ACM Special Interest Group on Multimedia (SIGMM)*, pages 139–148, 2011.

[3] M. A. Bartsch and G. H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7:96?104, 2005.

[4] Abdessalem Ben Abdelali, Mohamed Nidhalkrifa, Abdellatif Mtibaa, and El-Bay Bourennane. A study of color structure descriptor for shot boundary detection. *International Journal of Sciences and Techniques of Automatic Control and Computer Engineering*, 3(1):956–971, 2009.

[5] Sergio Benini, Aldo Bianchetti, Riccardo Leonardi, and Pierangelo Migliorati. Extraction of Significant Video Summaries by Dendrogram Analysis. In *Proceedings of the International Conference on Image processing (ICIP)*, pages 133–136, 2006.

[6] Jenny Benois-Pineau, William Dupuy, and Dominique Barba. Recovering of visual scenarios in movies by motion analysis and grouping spatio-temporal colour signatures of video shots. In *Proceedings of EUSFLAT'2001*, pages 385–389, 2001.

[7] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *The Tenth IEEE International Conference on Computer Vision (ICCV'05)*, pages 1395–1402, 2005.

[8] L. Cao, Z. Liu, and T.S. Huang. Cross-dataset action detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1998–2005, June 2010.

[9] M. Cuturi. Fast global alignment kernels. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 929–936, June 2011.

[10] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. *ECCV'06 Proc. of the 9th European conf. on Computer Vision - Volume Part II*, pages 428–441, 2006.

[11] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG '04, pages 253–262, New York, NY, USA, 2004. ACM. ISBN 1-58113-885-7. doi: 10.1145/997817.997857. URL `http://doi.acm.org/10.1145/997817.997857`.

[12] J.W. Davis and AF. Bobick. The representation and recognition of human movement using temporal templates. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 928–934, Jun 1997.

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

[14] Olivier Duchenne, Ivan Laptev, and Josef Sivic. Automatic annotation of human actions in video. In *Proc. of the Intl. Conf. on Computer Vision (2009)*, pages 1491–1498, 2009.

[15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd international conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[16] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis*, SCIA'03, pages 363–370, Berlin, Heidelberg, 2003. Springer-Verlag. ISBN 3-540-40601-8.

[17] A. Gaidon, Z. Harchaoui, and C. Schmid. Actom Sequence Models for Efficient Action Detection. *CVPR 2011 - IEEE Conference on Computer Vision & Pattern Recognition*, pages 3201–3208, June 2011.

[18] A. Gaidon, Z. Harchoui, and C. Schmid. A time series kernel for action recognition. *Proc. of the British Machine Vision conf. 2011*, pages 63.1–63.11, 2011.

[19] Andrey Goder and Vladimir Filkov. Consensus clustering algorithms: Comparison and refinement. In *Proceedings of 9th Workshop on Algorithm Engineering and Experiments (ALENEX'08)*, pages 109–117, 2008.

[20] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, November 1995. ISSN 0004-5411. doi: 10.1145/227683.227684. URL `http://doi.acm.org/10.1145/227683.227684`.

[21] Boqing Gong, Weil-Lun Chao, Kristen Grauman, and Fei Sha. Diverse Sequential Subset Selection for Supervised Video Summarization. In *Proceedings of the Neural Information Processing Systems Conference (NIPS)*, pages 1–9, 2014.

[22] N.A. Goussies, Z. Liu, and J. Yuan. Efficient search of top-k video subvolumes for multi-instance action detection. In *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, pages 328–333, July 2010.

[23] Jim Gray, Surajit Chaudhuri, Adam Bosworth, Andrew Layman, Don Reichart, Murali Venkatrao, Frank Pellow, and Hamid Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub totals. *Journal of Data Mining and Knowledge Discovery*, 1(1):29–53, 1997.

[24] M. C. Hughes and E. B. Sudderth. Nonparametric discovery of activity patterns from video collections. *IEEE Computer Vision & Pattern Recognition Workshops*, pages 25–32, June 2012.

[25] Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31:651–666, 2010.

[26] Prateek Jain, Sudheendra Vijayanarasimhan, and Kristen Grauman. Hashing hyperplane queries to near points with applications to large-scale active learning. In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 928–936. Curran Associates, Inc., 2010. URL `http://papers.nips.cc/paper/4088-hashing-hyperplane-queries-to-near-points-with-applications-to-large-scale-active-learning.pdf`.

[27] Xin Jin, Jiawei Han, Liangliang Cao, Jiebo Luo, Bolin Ding, and Cindy Kide Lin. Visual Cube and n-line analytical processing of images. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 849–858, 2010.

[28] A. Joly and O. Buisson. Random maximum margin hashing. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 873–880, June 2011. doi: 10.1109/CVPR.2011.5995709.

[29] Simon Jones, Ling Shao, Jianguo Zhang, and Yan Liu. Relevance feedback for real-world human action retrieval. *Pattern Recognition Letters*, 33(4):446 – 452, 2012. ISSN 0167-8655. Intelligent Multimedia Interactivity.

[30] Arijit Khan, Pouya Yanki, Bojana Dimcheva, and Donald Kossmann. Towards indexing functions: Answering scalar product queries. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 241–252, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2376-5. doi: 10.1145/2588555.2610493. URL `http://doi.acm.org/10.1145/2588555.2610493`.

[31] Youngdae Kim, Ilhwan Ko, Wook-Shin Han, and Hwanjo Yu. iKernel: Exact indexing for support vector machines. *Information Sciences*, 257(0):32–53, 2014. ISSN 0020-0255. doi: http://dx.doi.org/10.1016/j.ins.2013.09.017. URL `http://www.sciencedirect.com/science/article/pii/S0020025513006592`.

[32] A. Klaser, M. Marszalek, C. Schmid, and A. Zisserman. Human focused action localization in video. *Proceedings of the 11th European Conference on Trends and Topics in Computer Vision - Volume Part I*, pages 219–233, 2012.

[33] Yiannis Kompatsiaris, Bernard Merialdo, and Shiguo Lian, editors. *TV Content Analysis: Techniques and Applications*. CRC Press, 2012.

[34] I. Laptev. On space-time interest points. *Intl. Journal of Computer Vision*, 64(2-3):107–123, September 2005. ISSN 0920-5691.

[35] Ivan Laptev and Patrick Pérez. Retrieving actions in movies. *Proc. Int. Conf. on Computer Vision (ICCV'07)*, pages 1–8, Oct 2007.

[36] Yingbo Li and Bernard Merialdo. VERT: Automatic Evaluation of Video Summaries. In *Proceedings of ACM MultiMedia*, pages 851–854, 2010.

[37] Saloua Litayem, Alexis Joly, and Nozha Boujemaa. Hash-based support vector machines approximation for large scale prediction. In *Proceedings of the British Machine Vision Conference*, pages 86.1–86.11. BMVA Press, 2012. ISBN 1-901725-46-4. doi: http://dx.doi.org/10.5244/C.26.86.

[38] Wei Liu, Jun Wang, Yadong Mu, Sanjiv Kumar, and Shih-Fu Chang. Compact hyperplane hashing with bilinear functions. In *International Conference on Machine Learning (ICML)*, Edinburgh, Scotland, 2012.

[39] Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. Multi-probe LSH: Efficient indexing for high-dimensional similarity search. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pages 950–961. VLDB Endowment, 2007. ISBN 978-1-59593-649-3. URL `http://dl.acm.org/citation.cfm?id=1325851.1325958`.

[40] Benoit Mathieu, Slim Essid, Thomas Fillon, Jacques Prado, and Gal Richard. YAAFE, an Easy to Use and Efficient Audio Feature Extraction Software. In *Proceedings of the 11th International Society for Music Information Retrieval (ISMIR)*, page 441?446, 2010.

[41] Messing. The mpeg-7 color structure descriptor: Image description using color and local spatial information. In *Proceedings of the International Conference on Image processing (ICIP)*, pages 670–673, 2011.

[42] Huazhong Ning, T.X. Han, D.B. Walther, M. Liu, and T.S. Huang. Hierarchical space-time model enabling efficient search for human actions. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(6): 808–820, June 2009.

[43] A Oikonomopoulos, I Patras, and M. Pantic. Spatiotemporal localization and categorization of human actions in unsegmented image sequences. *Image Processing, IEEE Transactions on*, 20(4):1126–1140, April 2011. ISSN 1057-7149.

[44] D. Oneata, J. Verbeek, and C. Schmid. Action and Event Recognition with Fisher Vectors on a Compact Feature Set. In *ICCV 2013 - IEEE Intenational Conference on Computer Vision*, pages 1817–1824, Sydney, Australia, December 2013. IEEE.

[45] Dan Oneata, Jakob Verbeek, and Cordelia Schmid. Efficient Action Localization with Approximately Normalized Fisher Vectors. In *CVPR 2014 - IEEE Conference on Computer Vision & Pattern Recognition*, Columbus, OH, United States, June 2014. IEEE.

[46] P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, W. Kraaij, A. F. Smeaton, and G. Quénot. Trecvid 2013 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID 2013*. NIST, USA, 2013.

[47] Navneet Panda and E.Y. Chang. KDX: an indexer for support vector machines. *Knowledge and Data Engineering, IEEE Transactions on*, 18(6):748–763, June 2006. ISSN 1041-4347. doi: 10.1109/TKDE.2006.101.

[48] Vesa Peltonen, Juha Tuomi, Anssi Klapuri, Jyri Huopaniemi, and Timo Sorsa. Computational auditory scene recognition. In *Proceedings of the 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1941–1944, 2002.

[49] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. on PAMI*, 27(8):1226–1238, 2005.

[50] Florent Perronnin, Jorge Snchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. 6314:143–156, 2010. doi: 10.1007/978-3-642-15561-1_11. URL `http://dx.doi.org/10.1007/978-3-642-15561-1_11`.

[51] Julien Pinquier, Svebor Karaman, Laëtitia Letoupin, Patrice Guyot, Rémi Mégret, Jenny Benois-Pineau, Yann Gaëstel, and Jean-Franois Dartigues. Strategies for multiple feature fusion with Hierarchical HMM: Application to activity recognition from wearable audiovisual sensors. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR)*, pages 3192–3195, 2012.

[52] Karina R. Perez-Daniel, Mariko Nakano-Miyatake, Jenny Benois-Pineau, Sofian Maabout, and Gabriel Sargent. Scalable video summarization of cultural video documents in cross-media space based on data cube approach. In *Proceedings of the 12th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, 2014.

[53] M. Raptis and S. Soatto. Tracklet descriptors for action modeling and video analysis. *ECCV'10 Proc. of the 11th European conf. on Computer vision*, pages 577–590, 2010.

[54] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

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

[56] Ling Shao, S. Jones, and Xuelong Li. Efficient search and localization of human actions in video databases. *Circuits and Systems for Video Technology, IEEE Transactions on*, 24(3):504–512, March 2014.

[57] Le Song, Alex Smola, Arthur Gretton, Karsten M. Borgwardt, and Justin Bedo. Supervised feature selection via dependence estimation. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 823–830, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273600. URL `http://doi.acm.org/10.1145/1273496.1273600`.

[58] Andrei Stoian, Marin Ferecatu, Jenny Benois-Pineau, and Michel Crucianu. Fast cascaded action localization in video using frame alignment. *International Workshop on Computational Intelligence for Multimedia Understanding*, 2014.

[59] Y.L. Tian, L. Cao, Z. Liu, and Z. Zhang. Hierarchical filtered motion for action recognition in crowded videos. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 42(3):313–323, May 2012.

[60] Sudheendra Vijayanarasimhan, Prateek Jain, and Kristen Grauman. Hashing hyperplane queries to near points with applications to large-scale active learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(2):276–288, February 2014. ISSN 0162-8828. doi: 10.1109/TPAMI.2013.121. URL `http://dx.doi.org/10.1109/TPAMI.2013.121`.

[61] Paul Viola and Michael Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57 (2):137–154, 2002.

[62] H. Wang, A. Kläser, C. Schmid, and C.L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*, 103(1):60–79, May 2013.

[63] Heng Wang, Alexander Klser, Cordelia Schmid, and Cheng-Lin Liu. Dense trajectories and motion boundary descriptors for action recognition. *International Journal on Computer Vision*, 103:60–79, 2013.

[64] Jin Wang, Ping Liu, M.F.H. She, A.Z. Kouzani, and S. Nahavandi. The MPEG-7 color structure descriptor: Image description using color and local spatial information. In *Proceedings of 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2449–2454, 2011.

[65] Chuohao Yeo, P. Ahammad, K. Ramchandran, and S.S. Sastry. High-speed action recognition and localization in compressed domain videos. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(8):1006–1015, Aug 2008.

[66] M.M. Yeung and Boon-Lock Yeo. Time-constrained clustering for segmentation of video into story units. In *Proceedings of the 13th International Conference on Pattern Recognition (ICPR)*, volume 3, pages 375–380, 1996.

[67] Wen Yong-ge and Peng Sheng-ze. Research on image retrieval based on scalable color descriptor of mpeg7. *Advances in Control and Communications*, pages 91–98, 2012.

[68] G. Yu, J. Yuan, and Z. Liu. Unsupervised random forest indexing for fast action search. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 865–872, June 2011.

[69] G. Yu, J. Yuan, and Z. Liu. Real-time human action search using random forest based Hough voting. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 1149–1152, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0616-4.

[70] J. Yuan, Z. Liu, Y. Wu, and Z. Zhang. Speeding up spatio-temporal sliding-window search for efficient event detection in crowded videos. In *Proceedings of the 1st ACM International Workshop on Events in Multimedia*, EiMM '09, pages 3–8, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-754-7.

[71] J. Yuan, Z. Liu, and Y. Wu. Discriminative video pattern search for efficient action detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1728–1743, 2011. ISSN 0162-8828.

[72] Feng Zhou, Fernando De la Torre Frade, and Jessica K Hodgins. Aligned cluster analysis for temporal segmentation of human motion. In *IEEE Conference on Automatic Face and Gestures Recognition*, September 2008.