Modeling Video Evolution For Action Recognition

Basura Fernando, Efstratios Gavves, José Oramas M., Amir Ghodrati, Tinne Tuytelaars
KU Leuven, ESAT, PSI, iMinds
Leuven Belgium
firstname.lastname@esat.kuleuven.be

Abstract

In this paper we present a method to capture video-wide temporal information for action recognition. We postulate that a function capable of ordering the frames of a video temporally (based on the appearance) captures well the evolution of the appearance within the video. We learn such ranking functions per video via a ranking machine and use the parameters of these as a new video representation. The proposed method is easy to interpret and implement, fast to compute and effective in recognizing a wide variety of actions.

We perform a large number of evaluations on datasets for generic action recognition (Hollywood2 and HMDB51), fine-grained actions (MPII- cooking activities) and gestures (Chalearn). Results show that the proposed method brings an absolute improvement of 7-10%, while being compatible with and complementary to further improvements in appearance and local motion based methods.

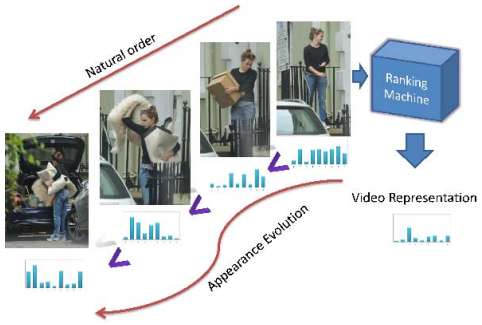

Figure 1: Illustration of how *VideoDarwin* works. In this video, as Emma *moved out from the house*, the appearance of the frames evolves with time. A ranking machine learns this evolution of the appearance over time and returns a ranking function. We use the parameters of this ranking function as a new video representation which captures vital information about the action.

1. Introduction

Most of the progress in the field of action recognition over the last decade has been related to i) the development of *local spatio-temporal descriptors* (going from spatio-temporal interest points [13], over dense sampling [14] to dense trajectories [37], and from gradient-based descriptors to motion-based [7] and motion-compensated ones), or ii) the adoption of powerful encoding schemes with an already proven track record in object recognition (e.g. Fisher Vectors [38]).

Modeling the *video-wide temporal evolution* of appearance in videos remains a challenging task, due to the large variability and complexity of video data. Actions are performed at largely varying speeds. Also the speed of the action often varies non-linearly within a single video. As a result, simple but robust techniques such as temporal pyramids (similar to the popular spatial pyramids [15] in object recognition) are insufficient. While several methods have been proposed to model the *video-wide temporal evolution* in actions (*e.g.* using HMM [39, 40], CRF-based methods [31] or deep networks [34]), the impact of these on action recognition performance so far has been somewhat disappointing. Nevertheless, it is clear that many actions have a characteristic temporal ordering - see e.g. the "moving out of the house" action in Figure 1. Intuitively, one would expect that a video representation that encodes this temporal ordering should help to better distinguish between different actions. In summary, obtaining a good video-wide representation remains a challenge.

In this paper, we approach a new video representation that captures this video-wide temporal evolution. We start from the observation that, even if the execution time of actions varies greatly, the *temporal ordering is typically preserved*. We propose to capture the temporal ordering *of a particular video* by training a linear ranking machine on the frames of that video. More precisely, given all the frames of the video, we learn how to arrange them in chronological order, based on the content of the frames. The parameters of the linear ranking functions encode the video-wide tempo-

ral evolution of appearance of videos in a principled way. To learn such ranking machines, we use the supervised learning to rank framework [17]. Ranking machines trained on different videos of the same action can be expected to have similar ranking functions. Therefore, we propose to use the parameters of the ranking machine as a new video representation for action recognition. Classifiers trained on this new representation turn out to be remarkably good at distinguishing actions. Since the ranking machines act on frame content (in our experiments local spatio-temporal descriptors), they actually capture both the appearance and their evolution over time. We call our method *VideoDarwin* .

Our key contribution is to use the parameters of the ranking functions as a new video representation that captures the *video-wide temporal evolution of the video*. Our new video representation is based on a principled learning approach, it is easy to implement and efficient. Last but not least, with the new representation we obtain state-of-the art results in action and gesture recognition.

The rest of the paper is organized as follows: in section 2 we position our work w.r.t. existing work. Section 3 describes our method. This is followed by the evaluation of our method in section 4. We conclude this paper in section 5.

2. Related work

Capturing temporal information of videos for action recognition has been a well studied research domain. Significant improvements have been witnessed in modeling local motion patterns present in short frame sequences [13, 37, 38]. Jain *et al*. [8] proposed to first localize the actions in the video and exploit them for refining recognition. To avoid using hand-engineered features, deep learning methodologies [16, 34] have also been investigated. Dynamics in deep networks can be captured either by extending the connectivity of the network architecture in time [11] or using stacked optical flow instead of frames as input for the network [29]. Although the aforementioned methods successfully capture the local changes within small time windows, they were not designed to model the higher level motion patterns and video-wide appearance/motion evolution associated with certain actions.

In modelling higher level motion patterns, machine learning approaches like CRF, HMM and action grammars, have been researched for action recognition [24, 27, 31, 33, 39]. In [39], a part-based approach is combined with large-scale template features to obtain a discriminative model based on max-margin hidden conditional random fields. In [31], Song *et al*. rely on a series of complex heuristics and define a feature function for the proposed CRF model. In [33] Tang *et al*. propose a max-margin method for modeling the temporal structure in a video. They use a HMM model to capture the transitions of action appearances and duration of actions.

Temporal ordering models have also been applied in the context of complex activity recognition [6, 26, 32]. They mainly focus on inferring composite activities from predefined, semantically meaningful, basic-level action detectors. In [32], a representation for events is presented that encodes statistical information of the atomic action transition probabilities using a HMM model. Similar to the above works, we exploit the temporal structure of videos but in contrast, we rely on ranking functions to capture the evolution of appearance or local motion. Using the learning to rank paradigm, we learn a functional representation for each video.

Due to the large variability of motion patterns in a video, usually latent sequential models are not efficient. To cope with this problem, representations in the form of temporal pyramids [4, 14] or sequences of histograms of visual features [3] are introduced. Different from them, we explicitly model video-wide, video level dynamics using a principled learning paradigm. Moreover, contrary to [3], our representation does not require manually annotated atomic action units during training. Recently, a study of encoding objects for actions and benefits of having objects in the video representation for action classification is presented in [9].

Our work has some conceptual similarity to function representations used in geometric modeling [20]. Since we use parameters of a linear function as a new representation, our work also has some similarity to the exemplar SVM concept [18], but our objective is to learn a representation for the relative ordering of a set of frames in a video. At the same-time we do not need to rely on negative data to learn the representation, as is the case for exemplar SVM.

3. *VideoDarwin* for action recognition

In this work the objective is to develop a video representation that captures the Video-wide Temporal Evolution (VTE) of videos. To achieve this, we consider a video as a vector valued function and learn to order frames-based appearance vectors or local-motion vectors. In section 3.1, we introduce the core idea of *VideoDarwin* . Then in section 3.2, we select a robust video vector valued function to capture the evolution of video content. The pipeline of *VideoDarwin* is illustrated in Figure 2.

3.1. Modeling Video-wide temporal evolution

We start from a video $X = [\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}]$ composed of n frames and frame at t is represented by vector $\mathbf{x_t} \in \mathbb{R}^D$. Based on the frame representations $\mathbf{x_t}$, we define a vector valued function V over the time variable t, $V : t \rightarrow \mathbf{v_t}$ where $\mathbf{v_t} \in \mathbb{R}^D$. The output of the vector valued function $\mathbf{v_t}$ is obtained by processing all the frames up to time t, denoted by $\mathbf{x_{1:t}}$. For example, the vector $\mathbf{v_t}$ can be obtained by applying the mean operation on all of

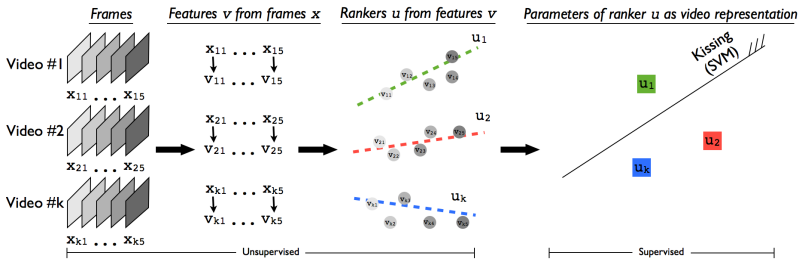

Figure 2: Processing steps of *VideoDarwin* for action recognition. First, we extract frames $\mathbf{x_1} \dots \mathbf{x_n}$ from each video. Then we generate feature $\mathbf{v_t}$ for frame t by processing frames from $\mathbf{x_1}$ to $\mathbf{x_t}$ as explained in section 3.2. Afterwards, using ranking machines we learn the video representation $\mathbf{u}$ for each video. Finally, video specific $\mathbf{u}$ vectors are used as a representation for action classification.

the frames $\mathbf{x_{1:t}}$. We discuss different ways to construct this vector valued function in section 3.2.

Due to the large variability in speed at which actions are performed, we propose to focus on relative orderings (i.e. $\mathbf{v_{t+1}}$ succeeds $\mathbf{v_t}$ which forms an ordering denoted by $\mathbf{v_{t+1}} \succ \mathbf{v_t}$). This way we end up with order constraints $\mathbf{v_n} \succ \dots \succ \mathbf{v_t} \succ \dots \succ \mathbf{v_1}$. We then *learn to order (rank)* the frames, for that particular video based on $\mathbf{v_t}$ ($t = \{1 \dots n\}$). The representation $V(t) = \mathbf{v_t}$ is chosen such that it is based on the frame-wise appearance or local motion information and only *indirectly* relates to the time variable t.

A natural way to model such order constraints is by the learning to rank paradigm [17], also known as ranking machines. Pairwise linear ranking machines learn a linear function characterized by the parameters $\mathbf{u} \in \mathbb{R}^D$, namely $\psi(\mathbf{v}; \mathbf{u}) = \mathbf{u}^T \cdot \mathbf{v}$. The ranking score of $\mathbf{v_t}$ is obtained by $\psi(\mathbf{v_t}; \mathbf{u}) = \mathbf{u}^T \cdot \mathbf{v_t}$ and results in the pairwise constraints $(\mathbf{v_{t+1}} \succ \mathbf{v_t})$ being satisfied by a large margin, while avoiding over-fitting. Namely, the learning to rank problem optimizes the parameters $\mathbf{u}$ of the function $\psi(\mathbf{v}; \mathbf{u})$, such that $\forall t_i, t_j, \mathbf{v_{t_i}} \succ \mathbf{v_{t_j}} \iff \mathbf{u}^T \cdot \mathbf{v_{t_i}} > \mathbf{u}^T \cdot \mathbf{v_{t_j}}$. Using the structural risk minimization and max-margin framework, the objective is then to optimize

$$\begin{aligned} \arg\min_{\mathbf{u}} \ & \frac{1}{2}\|\mathbf{u}\|^2 + C \sum_{\forall i,j \mathbf{v_{t_i}} \succ \mathbf{v_{t_j}}} \epsilon_{ij} \qquad (1) \\ s.t. \ & \mathbf{u}^T \cdot (\mathbf{v_{t_i}} - \mathbf{v_{t_j}}) \geq 1 - \epsilon_{ij} \\ & \epsilon_{ij} \geq 0. \end{aligned}$$

The linear ranking function ψ orders video data $\mathbf{v_t}$. As such, it captures the evolution of the video. Intuitively, the parameters $\mathbf{u}$ represent the information that frame representation $\mathbf{v_t}$ comes before the frame representation $\mathbf{v_{t+1}}$. Consequently, it learns how the frames evolve with regard to the appearance of the video. This appearance evolution information is encoded in parameter $\mathbf{u}$. In fact, the ranking parameters $\mathbf{u}$ can be viewed as a principled, data-driven, temporal pooling of the VTE.

Each video X_i has a different evolution of appearances over time and will therefore learn a different ranking function $\psi_i = \psi(\cdot; \mathbf{u_i})$. As the ranking function ψ_i is video specific, we propose to use the parameters $\mathbf{u_i} \in \mathbb{R}^D$ of ψ_i as a new video representation to capture the specific *appearance evolution of the video*. Thus we obtain a functional representation, where the functional parameters $\mathbf{u_i}$ of the ranking function ψ_i from eq. (1) serve as a representation that captures a vital part of the video-wide temporal information. We refer to our method as *VideoDarwin* since it exploits the evolution of appearance of a video.

Although the optimization objective is expressed on the basis of RankSVM [10], any other linear learning to rank method can be employed to learn *VideoDarwin* . So far the video wide temporal evolution of appearance captured by eq. (1) is linear. We incorporate non-linear families of functions by non-linear feature maps [36] applied on each $\mathbf{v_t}$ of V.

Generalization capacity. For action recognition we make the basic assumption that similar actions in different videos will have similar VTE. Namely, we assume there is a theoretical probability density function p_{VTE} based on which different instances of VTE are sampled for an action type. Naturally, different videos of the same action will generate different ranking functions and each linear ranker will have different parametric representation function ψ. Therefore, a rightful, question is to what extent learning the ψ per video generalizes well for different videos of the same action.

As we cannot know the theoretical probability density function p_{VTE} of VTE in real world videos, it is not possible to derive a strict bound on the generalization capacity of the functional parameters u_i. However, the sensitivity risk minimization framework gives us a hint of this generalization capacity of $\mathbf{u_i}$ when the input for the training is slightly perturbed. More specifically, Bousquet *et al.* [1] showed on a wide range of learning problems, *e.g.* SVM,

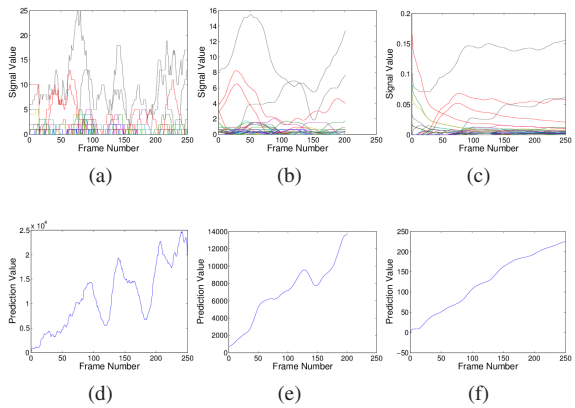

Figure 3: Vector value representations for *VideoDarwin* . For a random video, we see the signal for (a) the original independent frames, (b) moving average and (c) time varying mean vectors. Each colour represents a dimension. In (d), (e) and (f) y axis shows the predicted ranking score of each frame obtained from signal (a), (b) and (c) respectively after applying the ranking function (prediction ranking value at t $= \mathbf{u}^T \cdot \mathbf{v_t}$).

SVR and RankSVM, that the difference of the generalization risk R from the leave one out error $R^{/i}$ in the training set is bounded by

$$|R - R^{/i}| \leq E_r[|l(A_S, r) - l(A_{S^{/i}}, r)|] \leq \beta, \quad (2)$$

where A_S is a ranking algorithm with uniform stability β learned from the set of samples S. The expectation of the loss over the distribution r is denoted by $E_r[l]$ where l is a bounded loss function such that $0 \leq l(A_S, r) \leq M$; (M is a sufficiently small number).

Given a certain video, eq. (2) implies that a slight change (ignoring a single frame representation from the video vector valued function) during training will learn a ranking function $\psi_{/i}$ with an error no larger than β compared to the ψ learned when all frames are available. Although eq. (2) does not give a strict answer for what happens when the training input changes significantly from video to video, it hints that since the VTE of similar actions should be similar, this should also be the case for the learned ranking functions of *VideoDarwin* denoted by $\mathbf{u}$. This generalization capacity of *VideoDarwin* is furthermore supported by our experimental validation.

3.2. Vector valued functions for VideoDarwin

In this section we seek a good representation for the vector valued function, $V(t) = \mathbf{v_t}$. We discuss three methods to construct vector valued function $V(t)$ from frame data $\mathbf{x_t}$.

Independent Frame Representation. The most straightforward representation for capturing the evolution of the appearance of a video is to use independent frames as the output of the vector valued function $V(t) = \frac{\mathbf{x_t}}{\|\mathbf{x_t}\|}$. This approach might have two disadvantages. First, the original signal can be quite noisy (see Figure 3 (a)) and this would lead the ranking machines to learn undesirable noise. At the same time this strategy might generate ranking functions with high ranking errors during the training time. Secondly, independent frame representations result in a weak connection between $\mathbf{v_t}$ and t. Given this weak correlation between the $\mathbf{v_t}$ and time t, (see Figure. 3 (a)) the ranking function may not learn the appearance evolution over time properly. As a result, plotting the predicted score $s_t = \mathbf{u_i}^T \cdot \mathbf{v_t}$ for each of the frames in the video is not as smooth as one would desire (see Figure 3 (d)).

Moving Average (MA). Inspired by the time series analysis literature, we consider the moving average with a window size T as video representation at time t. In other words we consider locally smoothed signals.

For MA, we observe two facts; first, the output signal is much smoother (see Figure 3(b)) and secondly $\mathbf{v_t}$ depends locally, namely on the frames $[t, t+T]$. Unlike the independent frames representation, the moving average model forges a connection between $\mathbf{v_t}$ and t. Plotting these two variables for a window T=50 in Figure 3(b), we observe a smoother relation between the dimensions of $\mathbf{v_t}$ and the frame number (time variable). This is reflected in the global temporal information captured as well as in the predicted score s_t, (see Figure 3(e)). Although the moving average representation allows for capturing the appearance evolution of a video better, we still witness a general instability of the signals. Furthermore, the moving average representation introduces undesirable artifacts. First, a window size T must be chosen, which is not always straightforward as actions often take place in different tempos. Second, due to the dependency on the $[t, t+T]$ frames, naturally the last frames of a video are ignored. This is problematic, especially for videos of shorter duration.

Time Varying Mean Vectors. To deal with the limitations of the independent frames representation and the moving average, we propose to use the time varying mean vectors. Let us denote the mean at time t as $\mathbf{m_t} = \frac{1}{t} \times \sum_{\tau=1}^{t} \mathbf{x}_\tau$. Then, $\mathbf{v_t}$ captures only the direction of the unit mean appearance vector at time t, *i.e.* ($\mathbf{v_t} = \frac{\mathbf{m_t}}{||\mathbf{m_t}||}$). Thus the ranking function ψ learns the evolution of the normalized mean appearance at time t. We plot the relationship between $\mathbf{v_t}$ and t in Figure 3(c) and the prediction score s_t in Figure 3(f). We observe that, as desired, the output is smooth, almost resembling a monotonically increasing function. Different

from the independent frames representation, the time varying mean vectors introduce a better dependency between the input v_t and the target t.

By construction *time varying mean vectors* capture only the temporal information from the forward flow of the video with respect to the time (as the video progresses from the past to the future frames). However, there is no reason why the mean vectors should not be considered also in the reverse order, starting from the future frames and traversing backwards to the past frames of a video. To this end we generate the exact same objective, as in eq. 1, playing the video in reverse order, however. We shall refer to appearance evolution captured by forward flow as *Forward VideoDarwin* whereas reverse flow as *Reverse VideoDarwin* .

Video Classification with SVM. In order to capture VTE, we optimize for ψ_i by applying eq. (1) once for each video X_i separately. The vector $\mathbf{u_i}$ then represents a video specific evolution of appearance information for video X_i, see Figure 2 last column. Next, to classify videos into classes, we use a standard supervised classification method on our dataset of VTE representations denoted by $D_{train} = \{\mathbf{u_i}, y_i\}, i = 1, \ldots, N$ where N is the number of videos in our training set, y_i is the class label of the i^{th} video. We use non-linear SVM classifiers such as Chi-square feature maps [35] applied on feature vectors $\mathbf{u_i} \in \mathbb{R}^D$.

Finally, we summarize some of the advantages of *VideoDarwin* . No supervised information is needed as video order constraints can be obtained directly from the sequence of video frames. It captures the evolution of appearance of a video using a principled and well founded max-margin learning strategy. Learning to rank framework does not need any explicit negative data. As a result, *VideoDarwin* does not need any negative data during the representation learning stage. Since our method encapsulates the changes that happen in a video, it captures useful discriminative information for action recognition. It can be considered as a pooling strategy which we call the **rank pooling**.

4. Experiments

Now we present a detailed experimental evaluation of *VideoDarwin* .

Datasets. As the proposed methodology is not specific to an action type or class of actions, we present experiments in a broad range of datasets. We follow exactly the same experimental settings per dataset, using the same training and test splits and the same features as reported by the state-of-the-art methods.

HMDB51 dataset [12]. This is a generic action classification dataset composed of roughly 7,000 clips divided into 51 action classes. Videos and actions of this dataset are subject to different camera motions, viewpoints, video quality and occlusions. As done in the literature we use a one-vs-all classification strategy and report the mean classification accuracy over three standard splits provided by the authors in [12].

Hollywood2 dataset [14] This dataset has been collected from 69 different Hollywood movies that include 12 action classes. It contains 1,707 videos in total where 823 videos are used for training and 884 are used for testing. Training and test videos are selected from different movies. The performance is measured by mean average precision (mAP) over all classes, as in [14].

MPII cooking activities dataset for fine-grained action classification [25]. This dataset was created to evaluate fine-grained action classification. It is composed of 65 different actions that take place continuously within 8 hours of recordings. As the kitchen remains the same throughout the recordings, the classification focuses mainly on the content of the actions and cannot benefit from potentially discriminative background information (*e.g.* driving a car always takes place inside a car). We compute per class average precision using the same procedure as in [25] and report the final mAP.

ChaLearn Gesture Recognition dataset [2]. This dataset contains 23 hours of Kinect data of 27 persons performing 20 Italian gestures. The data includes RGB, depth, foreground segmentation and Kinect skeletons. The data is split into train, validation and test sets, with in total 955 videos each lasting 1 to 2 minutes and containing 8 to 20 non-continuous gestures. As done in the literature, we report precision, recall and F1-score measures on the validation set.

***VideoDarwin* and baselines.** In Sec. 4.1 and 4.2 we compare different variants of *VideoDarwin* . As a first baseline we use the state-of-the-art trajectory features (*i.e.* improved trajectories and dense trajectories) and pipelines as in [38, 37]. As this trajectory based baseline mainly considers *local temporal information* we refer to this baseline as **local**. We also compare with temporal pyramids (**TP**), by first splitting the video into two equal size sub-videos, then computing a representation for each of them like spatial pyramids [15]. For these baselines, at frame level we apply non-linear feature maps (*i.e.* power normalization for Fisher vectors and chi-squared kernel maps for bag-of-words based methods). We also compare different versions of *VideoDarwin* , we denote Forward *VideoDarwin* by **FDVD**, Reverse & Forward *VideoDarwin* by **RFDVD**, non-linear forward *VideoDarwin* by **NL-FDVD** and non-linear reverse & forward *VideoDarwin* by **NL-RFDVD**.

Implementation details. In principle there is no constraint on the type of linear ranking machines we employ for learning *VideoDarwin* . We have experimented with state-of-the-

art ranking implementation RankSVM [10] and SVR [30]. Both these methods can be used to solve learning to rank problems formulated in equation 1. We observe that both methods capture evolution of the video appearances equally well. As for SVR the learning convergence is notably faster, we will use the SVR solver of Lib-linear in this paper (C = 1).

For HMDB51 and Hollywood2 datasets we use state-of-the art improved trajectory features [38] with Fisher encoding [22]. As done in the literature, we extract HOG, HOF, MBH, and trajectory (TRJ) features from the videos. We create GMMs of size 256 after applying PCA with a dimensionality reduction of factor 0.5 on each descriptor. As done in [38], we also apply the square-root trick on all descriptors except for TRJ.

In order to compute non-linear *VideoDarwin* , we apply power normalization followed by a L2-normalization on individual Fisher vectors extracted from each video frame. For linear *VideoDarwin* , we just use Fisher vectors without any power normalization.

For MPII cooking dataset we use the features provided by the authors [25], that is bag-of-words histograms of size 4000 extracted from dense trajectory features [37] (HOG, HOF, MBH and TRJ). As we use bag-of-words for this dataset, in order to compute non-linear *VideoDarwin* , we apply χ^2-kernel maps on individual bag-of-words histograms after the construction of the vector valued function as explained in section 3.2.

For the ChaLearn Gesture Recognition dataset we start from the body joints estimated using the method from [28]. For each frame we calculate the relative location of each body joint w.r.t. the torso joint. Then, we scale these relative locations in the range [0,1]. We use a dictionary of 100 words to quantize these skeleton features. Similar to MPII cooking dataset, in order to compute non-linear *VideoDarwin* and for all baselines we use chi-squared kernel maps.

We train non-linear SVM classifiers with feature kernel maps for the final classification. Whenever we use bag-of-words representation we compute χ^2-kernel maps over the final video representation and then L2 normalize them. We use this strategy for both baselines and *VideoDarwin* . Similarly, when Fisher vectors are used, we apply power normalization and L2 normalization for the final video representation. The C parameter of SVM is cross-validated over the training set using two-fold cross-validation to optimize the final evaluation criteria (mAP, classification accuracy or F-score). When features are fused (combined) we use the average kernel strategy.

Execution time. *VideoDarwin* takes about 0.9 ± 0.1 sec per video on the Hollywood2 excluding the Fisher vector computation. The proposed algorithm is linear on the length of the video.

	HOG	HOF	MBH	TRJ.	Comb.	Comb. +local
Independent frames	41.6	52.1	54.4	43.0	57.4	64.1
Moving average (T=20)	42.2	54.6	56.6	44.4	59.5	64.6
Moving average (T=50)	42.2	55.9	58.1	46.0	60.8	65.0
Time varying mean vectors	**45.3**	**59.8**	**60.5**	**49.8**	**63.6**	**66.7**

Table 1: Comparison of different video representations for *VideoDarwin* . Results reported in mAP on the Holywood2 dataset using FDVD with Fisher vectors. As also motivated in Sec. 3.2, the time varying mean vector representation captures better the video-wide temporal information present in a video.

4.1. Evaluating representations for VideoDarwin

We first evaluate the three options presented in section 3.2 for the vector valued function, *i.e. independent frame*, *moving average* and *time varying mean vector* representations. We perform the experiments with Fisher vectors on the Hollywood2 dataset and summarize the results in Table 1. Similar trends were observed with dense trajectory features, bag-of-words and other datasets.

From the comparisons, we make several observations that validate our analysis. First, applying ranking functions directly on the Fisher vectors from the frame data captures only a moderate amount of the temporal information. Second, *moving average* applied with ranking seems to capture video-wide temporal information better than applying ranking functions directly on the frame data. However, the *time varying mean vector* consistently outperforms the other two representations by a considerable margin and for all features. We believe this is due to two reasons. First, *moving average* and *time varying mean vector* methods smooth the original signal. This reduces the noise in the signal. Therefore, it allows the ranking function to learn meaningful VTE. Secondly, the appearance information of the *time varying mean vectors* is more correlated with the time variable. The ranking function exploits this correlation to learn the evolution of the appearance over time in the video signal.

We conclude that *time varying mean vectors* are better for capturing the video-wide evolution of appearance of videos when applied with *VideoDarwin* . In the rest of the experiments we use the time varying mean vectors.

4.2. Action classification

Next, we present a detailed analysis of the action classification results in HMDB51, Hollywood2, MPII Cooking and ChaLearn Gesture recognition datasets (see Table 2, 3, 4 and 5 respectively).

VideoDarwin obtains better results in comparison to local temporal methods. Accurately modeling the evolution of appearance and motion, allows to capture more relevant

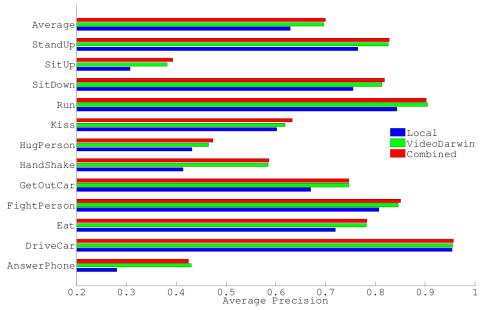

Figure 4: Per class AP in the Hollywood2 dataset. The AP is improved for all classes significantly, with an exception of "Drive car", where context already provides useful information.

	HOG	HOF	MBH	TRJ	Combined
Local	39.2	48.7	50.8	36.0	55.2
TP	40.7	52.2	53.5	37.0	57.2
FDVD	39.2	52.7	53.0	37.0	57.9
RFDVD	41.6	53.3	54.6	39.1	59.1
NL-FDVD	44.2	54.7	55.2	37.7	61.0
NL-RFDVD	**46.6**	**55.7**	**56.7**	**39.5**	**61.6**
Local + FDVD	42.4	53.7	54.3	39.7	59.3
Local + RFDVD	42.7	53.9	54.9	40.0	59.4
Local + NL-FDVD	45.6	56.2	56.2	41.0	61.3
Local + NL-RFDVD	**47.0**	**56.6**	**57.1**	**41.3**	**61.8**

Table 2: One-vs-all accuracy on HMDB51 dataset [12]

information for a particular action. In fact, the results verify our expectation, that what makes an action most discriminative from other actions is mostly the video-wide evolution of appearance and motion information of that action. The forward and reverse *VideoDarwin* reports consistent improvement over forward only *VideoDarwin* , further improved when non-linear *VideoDarwin* is employed. It is interesting to see that this trend can be observed in all four datasets too. Overall, local methods with *VideoDarwin* bring a substantial absolute increase over local methods ($+6.6\%$ for HMDB51, $+7.1\%$ for Holywood2, $+8.6\%$ for MPII Cooking, $+9.3\%$ for ChaLearn).

Analysis of action classification results. Looking at the individual results in the Hollywood2 dataset, see Fig. 4, we observe that almost all actions benefit the same, about a 7% average increase. Some notable exceptions are "answer phone", which improves by 14% and "handshake", which improves by 17%. For "drive car" there is no improvement. The most probable cause is that the car context already provides enough evidence for the classification of the action, also reflected in the high classification accuracy of the particular action. Our method improves over periodic actions such as "run, handshake" as well as non-periodic actions such as "get-out-of-car".

To gain further insight we, furthermore, investigate the mean similarity computed over classes on MPII cooking dataset with BOW-based MBH features in Figure 5. We

	HOG	HOF	MBH	TRJ	Combined
Local	47.8	59.2	61.5	51.2	62.9
TP	52.0	61.1	63.6	52.1	64.8
FDVD	45.3	59.8	60.5	49.8	63.6
RFDVD	50.5	63.6	65.5	**55.1**	67.9
NL-FDVD	52.8	60.8	62.9	50.2	65.6
NL-RFDVD	**56.7**	**64.7**	**66.9**	54.5	**69.6**
Local + FDVD	50.2	62.0	64.4	53.6	66.7
Local + RFDVD	52.7	64.3	66.2	55.9	68.7
Local + NL-FDVD	54.7	62.9	64.9	54.4	67.6
Local + NL-RFDVD	**57.4**	**65.2**	**67.3**	**56.1**	**70.0**

Table 3: Results in mAP on Hollywood2 dataset [19]

	HOG	HOF	MBH	TRJ	Combined
Local	49.4	52.9	57.5	50.2	63.4
TP	**55.2**	56.5	61.6	**54.6**	64.8
FDVD	50.7	53.5	58.0	48.8	62.4
RFDVD	53.1	55.2	61.4	51.9	63.5
NL-FDVD	52.8	**60.8**	**62.9**	50.2	**65.6**
NL-RFDVD	50.6	53.8	56.5	50.0	62.7
Local + FDVD	61.4	65.6	69.0	62.7	71.5
Local + RFDVD	**63.7**	**65.9**	**69.9**	**63.0**	71.7
Local + NL-FDVD	63.5	65.0	68.6	61.0	71.8
Local + NL-RFDVD	64.6	65.7	68.9	61.2	**72.0**

Table 4: Results in mAP on MPII Cooking fine grained action dataset [25].

	Precision	Recall	F-score
Local	65.9	66.0	65.9
TP	67.7	67.7	67.7
FDVD	60.6	60.4	60.5
RFDVD	65.5	65.1	65.3
NL-FDVD	69.5	69.4	69.4
NL-RFDVD	**74.0**	**73.8**	**73.9**
Local + FDVD	71.4	71.5	71.4
Local + RFDVD	73.9	73.8	73.8
Local + NL-FDVD	71.8	71.9	71.8
Local + NL-RFDVD	**75.3**	**75.1**	**75.2**

Table 5: Detailed analysis of precision and recall on the ChaLearn gesture recognition dataset [2]

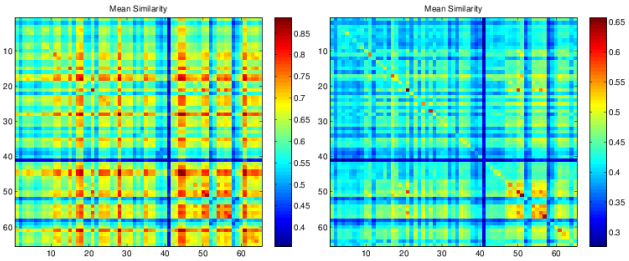

Figure 5: Mean class similarity obtained with (left) max-pooling and (right) *VideoDarwin* on MPII Cooking activities dataset using BOW-based MBH features extracted on dense trajectories. Non-linear forward *VideoDarwin* are used for our method.

construct the dot product kernel matrix using all the samples and then compute the mean similarity between classes, see Figure 5. The *VideoDarwin* kernel matrix (Figure 5 (right)) appears to be more discriminative than the one with max-pooled features (Figure 5 (left)). The action "smell"

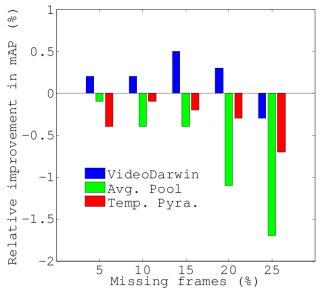

Figure 6: Comparison of action recognition performance after removing some frames from each video randomly on Hollywood2. *VideoDarwin* appears to be stable even when up to 20% of the frames are missing.

(#41) seems very difficult to discriminate either using max-pooling or *VideoDarwin* method. Actions "sneeze" (#44) and "stamp" (#45) seem to be very similar in-terms of appearances, however with *VideoDarwin* we can discriminated them better. Actions like "take & put in cupboard" (#47), "take & put in drawer"(#48), "take & put in fridge" (#49) and "take & put in oven" (#50) seem to be the most confused ones for *VideoDarwin* . These actions differ in the final instrument, but not in the dynamics of the action.

Stability analysis. We analyze the stability of *VideoDarwin* compared to average pooling and temporal pyramids. For this experiment we use Hollywood2 dataset and MBH features with Fisher vectors. We gradually remove 5%, 10%, . . . 25% of random frames from each video and then measure the change in mean average precision.

We present in Figure 6 the relative change in mAP after removing of frames. Typically, we would expect the mAP to decrease . Interestingly, with *VideoDarwin* removing up-to 20% of the frames from the video does not change the results significantly; in-fact we observe a slight relative improvement. This is a clear indication of the stability of *VideoDarwin* and an advantage of learning based temporal pooling. As expected, the mAP decreases for both average pooling method and the temporal pyramids method as the number of frames that are removed from videos increases. For average pooling mAP seems to drop almost in an exponential manner. However, it should be noted that 25% of the video frames is a significant amount of data. We believe the results outline the stability of *VideoDarwin* .

State-of-the-art and discussion. Last, we compare the nonlinear forward and reverse *VideoDarwin* combined with the local temporal information with the latest state-of-the-art in action recognition. We summarize the results in Table 6 and Table 7. Note that for Hollywood2 and HMDB51, we use the data augmentation by mirroring the videos as in [5].

By the comparisons of Tables 6 and 7, as well as from the results in the previous experiments, we draw several

	HMDB51	Hollywood2	Cooking
VideoDarwin	63.7	**73.7**	**72.0**
Hoai et al. *[5]*	60.8	73.6	–
Peng et al. *[21]*	**66.8**	–	–
Wu et al. *[42]*	56.4	–	–
Jain et al. *[7]*	52.1	62.5	–
Wang et al. *[38]*	57.2	64.3	–
Wang et al. *[37]*	46.6	58.2	–
Taylor et al. *[34]*	–	46.6	–
Zhou et al. *[44]*	–	–	70.5
Rohrbach et al. *[25]*	–	–	59.2

Table 6: Comparison of the proposed approach with the state-of-the-art methods sorted by reverse chronological order. Results reported in mAP for Hollywood2 and Cooking datasets. For HMDB51 we report one-vs-all classification accuracy.

	Precision	Recall	F-score
VideoDarwin	**75.3**	**75.1**	**75.2**
Pfister et al. *[23]*	61.2	62.3	61.7
Yao et al. *[43]*	–	–	56.0
Wu et al. *[41]*	59.9	59.3	59.6

Table 7: Comparison of the proposed approach with the state-of-the-art methods on ChaLearn gesture recognition dataset sorted by reverse chronological order.

conclusions. First, *VideoDarwin* is useful to encode video-wide, temporal information. Also, *VideoDarwin* is complementary to action recognition methods that compute local temporal features, such as improved trajectory-based features [38]. In fact, fusing *VideoDarwin* with the previous state-of-the-art in local motion and appearance, we improve up to 10%. *VideoDarwin* is only outperformed on HMDB51 by [21], who combine their second layer Fisher vector features with normal Fisher vectors to arrive at 205K dimensional vectors and a 66.8% accuracy. When using Fisher vectors like *VideoDarwin* does, Peng *et al*. [21] obtain 56.2%, which is 10% lower than what we obtain with *VideoDarwin* .

5. Conclusion

We introduce *VideoDarwin* , a method that models the evolution of appearance and motion information in a video. *VideoDarwin* is an unsupervised, learning based temporal pooling method, which aggregates the relevant information throughout a video via a learning to rank methodology. The generalization properties, as well as the regularized learning, of the learning to rank algorithms allow us to capture the global temporal information of a video while minimizing the empirical risk. As a result we arrive at a robust video representation suitable for action recognition. Based on extensive experimental evaluations on different datasets and features we conclude that, our method is applicable to any frame-based representations for capturing the global temporal information of a video.

Acknowledgment

The authors acknowledge the support of the EC FP7 project AXES, FP7 ERC Starting Grant 240530 COGNIMUND, DBOF PhD scholarship, the FWO project *Monitoring of abnormal activity with camera systems* and iMinds High-Tech Visualization project.

References

[1] O. Bousquet and A. Elisseeff. Stability and generalization. *JMLR*, 2:499–526, 2002. 3

[2] S. Escalera, J. Gonzàlez, X. Baró, M. Reyes, O. Lopes, I. Guyon, V. Athitsos, and H. Escalante. Multi-modal gesture recognition challenge 2013: Dataset and results. In *ICMI*, 2013. 5, 7

[3] A. Gaidon, Z. Harchaoui, and C. Schmid. Actom sequence models for efficient action detection. In *CVPR*, 2011. 2

[4] A. Gaidon, Z. Harchaoui, C. Schmid, et al. Recognizing activities with cluster-trees of tracklets. In *BMVC*, 2012. 2

[5] M. Hoai and A. Zisserman. Improving human action recognition using score distribution and ranking. In *ACCV*, 2014. 8

[6] H. Izadinia and M. Shah. Recognizing complex events using large margin joint low-level event model. In *ECCV*, 2012. 2

[7] M. Jain, H. Jégou, and P. Bouthemy. Better exploiting motion for better action recognition. In *CVPR*, 2013. 1, 8

[8] M. Jain, J. van Gemert, H. Jegou, P. Bouthemy, and C. G. Snoek. Action localization with tubelets from motion. In *CVPR*, 2014. 2

[9] M. Jain, J. van Gemert, and C. G. M. Snoek. What do 15,000 object categories tell us about classifying and localizing actions? In *CVPR*, 2015. 2

[10] T. Joachims. Training linear svms in linear time. In *ICKDD*, 2006. 3, 6

[11] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 2

[12] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, 2011. 5, 7

[13] I. Laptev. On space-time interest points. *IJCV*, 64:107–123, 2005. 1, 2

[14] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008. 1, 2, 5

[15] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006. 1, 5

[16] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011. 2

[17] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009. 2, 3

[18] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 89–96. IEEE, 2011. 2

[19] M. Marszałek, I. Laptev, and C. Schmid. Actions in context. In *CVPR*, 2009. 7

[20] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko. Function representation in geometric modeling: concepts, implementation and applications. *The Visual Computer*, 11(8):429–446, 1995. 2

[21] X. Peng, C. Zou, Y. Qiao, and Q. Peng. Action recognition with stacked fisher vectors. In *ECCV*, 2014. 8

[22] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *CVPR*, 2010. 6

[23] T. Pfister, J. Charles, and A. Zisserman. Domain-adaptive discriminative one-shot learning of gestures. In *ECCV*, 2014. 8

[24] M. Raptis, I. Kokkinos, and S. Soatto. Discovering discriminative action parts from mid-level video representations. In *CVPR*, 2012. 2

[25] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *CVPR*, 2012. 5, 6, 7, 8

[26] M. Rohrbach, M. Regneri, M. Andriluka, S. Amin, M. Pinkal, and B. Schiele. Script data for attribute-based recognition of composite activities. In *ECCV*, 2012. 2

[27] M. S. Ryoo and J. K. Aggarwal. Recognition of composite human activities through context-free grammar based representation. In *CVPR*, 2006. 2

[28] J. Shotton, A. W. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R.Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, 2011. 6

[29] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. *CoRR*, abs/1406.2199:1–8, 2014. 2

[30] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14:199–222, 2004. 6

[31] Y. Song, L.-P. Morency, and R. Davis. Action recognition by hierarchical sequence summarization. In *CVPR*, 2013. 1, 2

[32] C. Sun and R. Nevatia. Active: Activity concept transitions in video event classification. In *ICCV*, 2013. 2

[33] K. Tang, L. Fei-Fei, and D. Koller. Learning latent temporal structure for complex event detection. In *CVPR*, 2012. 2

[34] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Convolutional learning of spatio-temporal features. In *ECCV*, 2010. 1, 2, 8

[35] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008. 5

[36] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *PAMI*, 34:480–492, 2012. 3

[37] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *IJCV*, 103:60–79, 2013. 1, 2, 5, 6, 8

[38] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013. 1, 2, 5, 6, 8

[39] Y. Wang and G. Mori. Hidden part models for human action recognition: Probabilistic versus max margin. *PAMI*, 33:1310–1323, 2011. 1, 2

[40] D. Wu and L. Shao. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In *CVPR*, 2014. 1

[41] J. Wu, J. Cheng, C. Zhao, and H. Lu. Fusing multi-modal features for gesture recognition. In *ICMI*, 2013. 8

[42] J. Wu, Y. Zhang, and W. Lin. Towards good practices for action video encoding. In *CVPR*, 2014. 8

[43] A. Yao, L. Van Gool, and P. Kohli. Gesture recognition portfolios for personalization. In *CVPR*, 2014. 8

[44] Y. Zhou, B. Ni, S. Yan, P. Moulin, and Q. Tian. Pipelining localized semantic features for fine-grained action recognition. In *ECCV*, 2014. 8