Training Initialization of Hidden Markov Models in Human Action Recognition

Zia Moghaddam and Massimo Piccardi, *Senior Member, IEEE*

***Abstract*—Human action recognition in video is often approached by means of sequential probabilistic models as they offer a natural match to the temporal dimension of the actions. However, effective estimation of the models' parameters is critical if one wants to achieve significant recognition accuracy. Parameter estimation is typically performed over a set of training data by maximizing objective functions such as the data likelihood or the conditional likelihood. However, such functions are non-convex in nature and subject to local maxima. This problem is major since any solution algorithm (expectation-maximization, gradient ascent, variational methods and others) requires an arbitrary initialization and can only find a corresponding local maximum. Exhaustive search is otherwise impossible since the number of local maxima is unknown.**

While no theoretical solutions are available for this problem, the only practicable mollification is to repeat training with different initializations until satisfactory cross-validation accuracy is attained. Such a process is overall empirical and highly time-consuming. In this paper, we propose two methods for one-off initialization of hidden Markov models achieving interesting trade-offs between accuracy and training time. Experiments over three challenging human action video datasets (Weizmann, MuHAVi and Hollywood Human Actions) and with various feature sets measured from the frames (STIP descriptors, projection histograms, notable contour points) prove that the proposed one-off initializations are capable of achieving accuracy above the average of repeated random initializations and comparable to the best. In addition, the methods proposed are not restricted solely to human action recognition as they suit time series classification as a general problem.

***Notes to Practitioners*—This paper presents a method useful for the development of automated video surveillance systems. The current generation of video surveillance systems is geared towards recognizing actions of interest (such as tampering with infrastructure, drawing graffiti, looking inside a car etc) automatically from video streams. To this aim, computer algorithms can be employed to learn to recognize such actions simply from presentation of example videos. However, the learning time can prove excruciatingly high, up to the order of days or weeks of computing time depending on the size of the dataset. This paper presents two methods to dramatically reduce such a learning time. The main practical advantage is the rapid re-configurability of the video surveillance system to recognize new and different action types arising from different scenarios of application.**

Manuscript submitted on November 29, 2012.

Zia Moghaddam and Massimo Piccardi are with the Faculty of Engineering and Information Technology, University of Technology, Sydney (UTS), Australia (phone: 61-2-95147942; e-mail: Massimo.Piccardi@uts.edu.au).

***Index Terms*—HMM parameter initialization, maximum likelihood estimation, expectation-maximization, human action recognition.**

I. Introduction

AUTOMATIC recognition of human actions in video allows automation of many otherwise manually-intensive tasks such as video surveillance, retrieval of videos from large databases, pedestrian traffic monitoring, and many others. Understanding human behavior is a high-level task relying on several, lower-level tasks such as segmentation, tracking and pose estimation. The typical goal of automatic action recognition is the classification of a given image sequence as one of several classes of pre-defined actions.

Many different approaches for action recognition have been proposed over the past two decades. The most recent surveys in [1-3] offer a comprehensive overview. However, it is a common opinion that many open issues still affect the accuracy of action recognition. As a main challenge, the instances of the same action by various people are significantly different; moreover, every individual performs each action in a different manner over various instances, both in space and time. This can be formulated as a problem of high, intrinsic within-class variability. Adding to the challenge, the number of samples available for training is typically limited compared to the parameters, preventing a "brute force" training approach.

Recognition of human actions typically requires the classification of a time series of measurements, also called observations, $O = \{o_1, \dots, o_t, \dots o_T\}$, extracted from the video depicting the action. Amongst the common approaches to classification, the hidden Markov model (HMM) [4] has played a special role in that it has been widely used and has inspired a number of other similar *probabilistic graphical models*, both directed and undirected (hidden semi-Markov models, layered HMM, conditional random fields, and many others [5-7]). However, certain problems with the training of graphical models are still partially unresolved. The main principle guiding the training of a model from a set of observation sequences, $O^e = \{o_1^e, \dots, o_t^e, \dots o_{T_e}^e\}$, e = 1...E, is to learn its set of parameters, noted as λ hereafter, with maximum likelihood. The likelihood function contains hidden variables and is therefore non-convex in nature, giving lieu to an undefined number of local maxima. Solution algorithms such as gradient ascent, expectation-maximization (EM) [8],

variational methods [9], are only guaranteed to converge to a local maximum of the likelihood which, in turn, directly depends on an arbitrary initialization of the training algorithm. This problem is major as the difference in value of such maxima can be extreme, with corresponding major changes in the accuracy of the recognition task. The main workaround for this problem is that of repeating the training stage many times from different initializations until satisfactory parameters are found.

For these reasons, this paper investigates and presents a number of approaches that can improve the training time of HMMs with one-off initializations and apply them to the recognition of human actions in video. The advantage brought by one-off initialization is significant as training is generally time-consuming and has also to be repeated many times over various experimental setups and datasets. For instance, a single training session over 400 training sequences with a 2.8 GHz CPU PC and Matlab 7.9 takes us between 8 and 36 hours depending on the model's complexity. In the case of larger training sets, the training time is in the order of weeks and repeating the training is heavily time-consuming. Therefore, one-off initialization approaches remove practical obstacles to the optimal tuning of models. The main contributions of this paper can be summarized as follows:

- a novel method for initializing HMM training based on time segmentation of the training sequences (Section IV.B);
- a novel method for initializing HMM training based on an incrementally-constructed histogram of the training sequences (Section IV.B);
- a novel method for assigning components to mixture distributions and its use in HMM training (Section IV.C);
- a heuristic indicator of optimal initialization based on the weighted Kullback-Leibler divergence (Section V.C).

The rest of this paper is organized as follows: Section II offers a brief review of the main related work on training initialization. Section III summarizes the HMM to the extent required to understand the methods presented in the paper. In Section IV, we present the proposed HMM initialization strategies. Experimental results are reported and discussed in Section V. Finally, conclusions are drawn in the last section.

II. Related work

Any action recognition approach requires the extraction of informative measurements from the video and their ensuing classification. Various lines of investigations have been followed in human action recognition research during the last two decades. In terms of classification approaches, three main approaches have been adopted: 1) recognizing the action directly in the time domain (e.g., [10, 11]); 2) recognizing the action by graphical models [6, 12, 13]; and 3) recognizing the action by classification of histograms of the measurements [14-16]. The time domain approach has dynamic time warping (DTW) [10] as its main representative, while the hidden Markov model (HMM) [17-19] is the reference generative approach for graphical models. Other graphical models such as dynamic Bayesian networks and conditional random fields have also been used with significant degree of success (e.g. [1, 7]). The approaches based on classification of histograms convert the time series of measurements, which varies in length for every action instance, into a histogram of values and then apply any conventional classifier such as the support vector machine or nearest-neighbors for the classification stage. Such histogram-based approaches have reported significant empirical accuracy [14-16]: however, it can be argued that they do not appropriately factor in the temporal dimension of human actions. In contrast, sequential classifiers such as HMM can naturally classify sequences of arbitrary length and have shown good performance in adjusting to variations in the duration of instances of the same action. In addition, a generative model like HMM can be more easily combined into complex, hierarchical models as shown, amongst others, by the work from Ikizler and Forsyth [20]. For this reason, we focus on HMM in the following, yet providing extensive comparison with other approaches.

Each measurement extracted from the video is modeled as a multivariate random variable of given dimensionality and is commonly referred to as *feature set* or *feature vector*. The feature sets used for action recognition in the literature can be categorized into two main groups: global features and local features [1]. The global features represent the body of an individual in a holistic manner, while the local features are a collection of local patches that are centered on salient points in space or space-time domain. Local features enjoy some invariance to viewpoint changes and are more robust to occlusions. Popular local features are the histogram of oriented gradients (HOG) and the histogram of oriented flow (HOF) collected at space-time interest points (STIP) [21]. At the two extremes, approaches can be based on either rich, general-purpose feature vectors, possibly invariant to the viewpoint (e.g. [22]), or minimal, fast-to-extract feature vectors optimized for specific types of actions [23, 24].

Since the training of graphical models with maximum likelihood is notoriously dependent on the initial parameter assignment, previous research has focused on ways to mitigate such a dependency. For instance, Ferrer *et al*. in [25] reviewed various HMM initializations based on random techniques and introduced a new method based on an equal-state frequency criterion. The idea is to first generate a set of HMM parameters as a sample from an arbitrary prior distribution. As next step, the Viterbi decoding algorithm is run on all the training sequences and the number of occurrences of each state is recorded. If such numbers are approximately equal, the parameters' sample is retained as the initial parameters of maximum-likelihood training. Otherwise, a new sample is generated until this condition is met. While this method is based on a reasonable principle, the average number of samples to be generated until the stop condition is met may prove unworkable, especially for HMM with many states. In [26], Liu *et al.* proposed an HMM initialization method for

gesture recognition application which computes directly the HMM initial parameters through simple physical considerations. In their work, the observation sequences of human gestures are evenly segmented into HMM states. This assumption of linear correspondence between time and states is plausible for some applications and we exploit it for one of the initialization methods proposed in this paper. In [27], Toledano *et al*. have explored three different ways of initializing HMM: 1) by a fixed template for all classes; 2) by historical averages; and 3) by oracle initialization (this last only to establish offline upper bounds). While these methods remove undesired randomness, they do not adapt the initial model to the specific training data. In a recent paper [28], Aillón Clemente *et al*. have proposed a sophisticated adaptive initialization scheme based on multiple sequence alignment. Yet, their method is specific to the HMM with left-to-right topology which is a more restricted case than that considered in this paper. Overall, the prevalent approach for training initialization remains that of running multiple training sessions with different random initializations of λ until satisfactory parameters are found, in terms of either a sufficiently-high likelihood or sufficient accuracy from a cross-validation experiment. The second target is usually much preferable, given that maximizing the likelihood alone is prone to over-fitting. However, such random trials can prove very time-consuming and have no formal stop criterion. Therefore, in this paper we propose various one-off initialization methods, adaptive to the actual training data.

III. HMM AND ACTION CLASSIFICATION

The HMM is a probabilistic graphical model over a sequence of observations, $O = \{o_1,...,o_t,...,o_T\}$, and a sequence of corresponding hidden states, $Q = \{q_1,...,q_t,...,q_T\}$ [4]. Each state takes values in a discrete set of N symbols, $S = \{s_1,...,s_N\}$, while the observations can be of either discrete or continuous nature. The HMM posits a simplified joint probability for O and Q that factorizes as:

$$p(O,Q) = p(q_1)\prod_{t=2}^{T} p(q_t|q_{t-1})\prod_{t=1}^{T} p(o_t|q_t) =: \pi_{q_1}\prod_{t=2}^{T} a_{q_{t-1},q_t}\prod_{t=1}^{T} b_{q_t}(o_t)\,. \quad (1)$$

where term $p(q_1)$ is called the initial state probability, while terms $p(q_t \mid q_{t-1})$ and $p(o_t \mid q_t)$ are known as state transition probabilities and observation probabilities, respectively. For notational convenience, these quantities are often simply noted as $\pi_{q_1}, a_{q_{t-1},q_t}, b_{q_t}(o_t)$, following [4]. Given (1), the HMM can be fully identified by its parameter set, λ:

$$\lambda = \{A, B, \pi\}, \quad A = \{a_{ij}\}, \quad B = \{b_i(o)\}, \quad \pi = \{\pi_i\} \quad i,j = 1 \dots N \quad (2)$$

where A is the N x N state transition probability matrix, B stands for the parameters of the observation probabilities and π are the N x 1 initial-state probabilities. In our case, the observations are continuous, multivariate random variables and their distribution for each state s_i, i=1...N, is modeled as a Gaussian mixture model (GMM) with M components:

$$b_i(o) := p(o|q_t = s_i) = \sum_{l=1}^{M} c_{il}\,\mathcal{N}(o|\mu_{il}, \Sigma_{il}) \quad (3)$$

where μ_{il} and Σ_{il} are the mean and covariance of the l-th Gaussian component and c_{il} is its weight in the mixture. Hence, the total size of B is ($N * M * (|\mu_{il}| + |\Sigma_{il}| + |c_{il}|)$), where by the cardinality operator | | we mean the number of elements of the argument. Such a number is typically high and confirms that an HMM is a highly parametric model.

A. Action Classification with HMM

The HMM can be used to compute the probability of an observation sequence in problems of action classification. Let us call C a set of K action classes, $C = \{c_1,\dots,c_k,\dots,c_K\}$, with a corresponding HMM for each class, c_k, k = 1...K, noted by its set of parameters, λ_k. Any action instance, O, can be classified into a class by using a maximum-likelihood classification rule:

$$c^* = argmax_{k=1\dots K}\big(p(O|\lambda_k)\big) \quad (4)$$

where $p(O|\lambda_k)$ is the likelihood of sequence O in the k-th class/model, efficiently computable by marginalization of states Q in (1) via the forward-backward algorithm [4]. If desired, priors and costs can be easily added to (4) to extend the classification criterion from maximum-likelihood to maximum-a-posteriori or minimum expected risk.

B. HMM Training

The HMM parameters are commonly estimated by fitting the model to a training set of observation sequences, $O^e = \{o_1^e, \dots, o_t^e, \dots o_{T_e}^e\}$, e = 1...E, with maximum likelihood [4]:

$$\lambda^* = argmax_\lambda\left(\prod_{e=1}^{E} p(O^e|\lambda)\right) \quad (5)$$

Training is unsupervised with respect to the states, i.e. states Q^e are treated as unobserved variables, causing multiple maxima in the likelihood function in (5). In addition, (5) is often expressed in a logarithmic scale (log-likelihood) to replace the product by a sum. The most popular HMM training algorithm is *Baum-Welch* which obtains a maximum for the log-likelihood by iterative maximization of a lower bound [4]:

$$\lambda^{i+1} = = argmax_\lambda\left(\sum_{e=1}^{E}\sum_{Q^e} \ln p(O^e, Q^e|\lambda)\, p(Q^e|O^e, \lambda^i)\right) \quad (6)$$

The expectation in (6) can be proved a lower bound for the log-likelihood and therefore by maximizing it over λ (i.e., the model), we are assured to select a model of at least equivalent log-likelihood. The maximization in (6) must be repeated iteratively, with parameter λ^i in the (i+1)-th iteration set to be the model returned by the previous, i-th iteration. This iterative scheme is guaranteed convergence to a local maximum (or a saddle point) of the log-likelihood, and the position and quality of this maximum depend heavily on the arbitrary parameters, λ^0, that were used for the first iteration, justifying the objectives of this work.

IV. Initialization of HMM Training

While all of the HMM's initial parameters influence the outcome of training, in the following we focus only on the parameters of the observation probabilities, B, because their size is typically overwhelming. For instance, in an HMM with N = 5 states, M = 5 components per mixture, F = 10 dimensions for the multivariate observations (a conservative figure) and full covariance matrices, the size of B is equivalent to 1,645 scalar parameters.

A. Random Initialization Method

Before the proposed initialization approaches, we illustrate a conventional method that will be used as reference in the rest of this paper [4, 29]. As noted, parameter B consists of N sets of M weighted Gaussian components, $\{\mu_{il}, \Sigma_{il}, c_{il}\}$, i = 1...N, l = 1...M. The conventional method obtains such values in the following two steps (Fig. 2, top of next page):

Cluster initialization: in the first step, all the observations from all training instances, O^e, e=1...E, are merged into a single set that we refer to as the *super vector*. Then, a clustering algorithm - k-means - is used to partition the observations into (N*M) clusters which are used to initialize B as follows: the μ_{il} means are set as the clusters' centers; the Σ_{il} covariances are set as the sample covariances in each cluster, and the c_{il} weights are set proportionally to the number of samples in each cluster.

Unfortunately, the k-means algorithm requires an arbitrary initialization at its turn. A k-means algorithm is very much alike a simplified EM algorithm: it is iterative in nature; it requires an arbitrary, initial clustering of the data to start from; and it is guaranteed to converge to more compact clusters than the starting ones [30]. In [4, 29], the required (N*M) initial centers are chosen randomly with uniform probability from the data in the super vector. In the following, we refer to this initialization method as *random centers*.

Component dispatching: as the second step of initialization, the (N*M) resulting components are dispatched over the N states (M components to each state) in *appearance order*, and used as the starting point of EM training.

In the following two sub-sections (IV.B and IV.C), we propose two one-off initialization methods designed to amend this trial-and-error style of initialization. Albeit heuristic in nature, the proposed methods are well founded in the temporal and spatial dimensions of the time series.

B. Proposed Methods: One-Off Initialization

The sequential observation data in an HMM with GMM observation densities are modeled based on two dimensions:

1. *Sequentiality* (or proximity in time): states have a natural duration expressed by the transition probability matrix. Observations which are close in time are more likely to belong to the same state and, therefore, being drawn from the same GMM.

2. *Proximity* in feature space: each observation has a value, irrespectively of its occurrence in time. In general, those observations whose values are close to each other are more likely to be generated from the same state.

Fig. 1 illustrates these concepts by plotting the observations from an instance of action "ClimbLadder" in the MuHAVi human action dataset [31]. Each observation is a multivariate measurement encoding the pose of the actor in one video frame. Fig 1.a shows two dimensions of the observations (dimensions 3 and 4) against the time. The progression of the observation values seemingly justifies the assumption that they could be generated from a sequence of discrete states such as those of an HMM. Fig. 1.b shows the same observations as a scatter plot after removing the time dimension. The plot shows how the observations tend to aggregate into clusters which can therefore be ascribed to the emissions of single states.

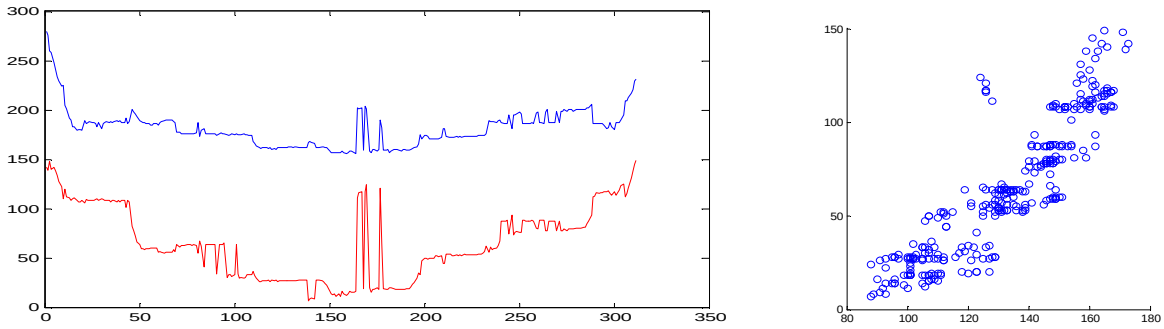

Fig. 1. Plots of the observations for an instance of action "ClimbLadder" from the MuHAVi action dataset [31]: a) dimensions 3 and 4 of the observations against the time (40 ms units); b) the same dimensions as a scatter plot.

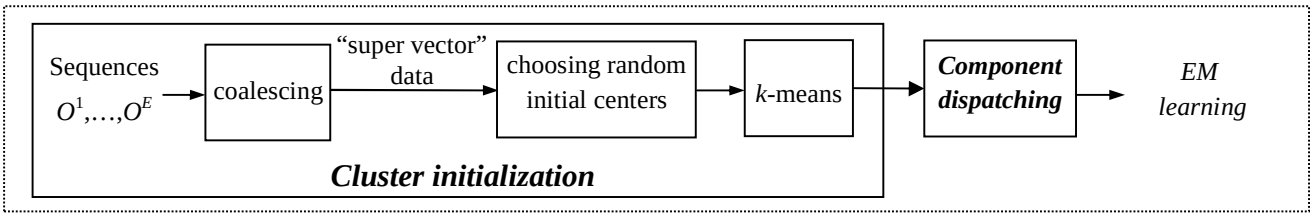

Fig. 2: HMM initialization using random initial centers.

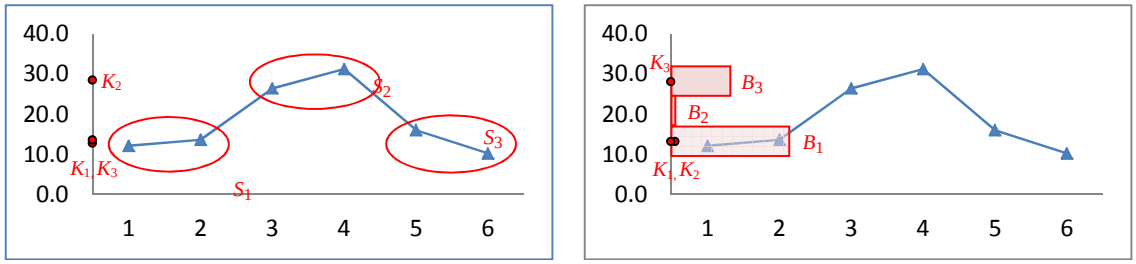

Fig. 3: An illustrative example: a) time-based initialization; b) histogram-based initialization.

The role of HMM training is that of balancing the two dimensions of sequentiality and proximity. However, the trade-off is not known at initialization time and the natural choice is therefore to use either dimension to develop an initialization strategy. Accordingly, in this paper we propose the two following cluster initialization approaches: 1) a "*time-based*" approach, exploiting sequentiality, and 2) a "*histogram-based*" approach, exploiting proximity in feature space. Unlike random initialization of clusters' centers, both methods permit one-off training of the HMM with significant reduction of the training time. Before delving with the formal description of these initializations, we illustrate them by a "toy" numerical example. Let us assume that we have a short sequence of six one-dimensional observations, O = {12.1, 13.6, 26.4, 31.2, 16.0, 10.2}, that we wish to explain by an HMM with three states and unimodal emissions; thus, N = 3 and M = 1. This requires us to choose N*M = 3 centers, noted as K_1, K_2 and K_3, for the initialization of the *k*-means algorithm.

With the time-based approach, the sequence gets evenly divided into three sub-sequences, S_1 = {12.1, 13.6}, S_2 = {26.4, 31.2}, and S_3 = {16.0, 10.2}. Then, the initial centers are set as the average of the data in each sub-sequence: K_1 = 12.9, K_2 = 28.8 and K_3 = 13.1. Fig. 3.a shows this case.

Conversely, with the histogram-based approach, the interval spanned by the data, [10.2, 31.2], is evenly divided into three bins of width 7.0 each:

- B_1: range = [10.2, 17.2], number of data in bin = 4, average value = 13.0;
- B_2: range = [17.2, 24.2] number of data in bin = 0, average value = 0;
- B_3: range = [24.2, 31.2] number of data in bin = 2, average value = 28.8.

The initial centers are then set as the bins' data averages, repeating the same value multiple times proportionally to the share of data contained in each bin. This results in: K_1 = 13.0, K_2 =13.0 and K_3 = 28.8. In turn, Fig. 3.b shows this case. The formal descriptions are presented hereafter.

1) Time-Based Initialization

In the time-based approach, we initialize the clusters' centers by partitioning the observation data in the time domain (Fig. 5.a, top of next page). In this method (named *average of training instances* hereafter), we first partition the frame sequence of each training instance, O^e, into (N*M) consecutive segments, $\{S_p^e\}_{p=1}^{N*M}$, each of equal length, $\lfloor T_e / (N*M) \rfloor$. Then, for each S_p^e segment, the mean value of its observations is taken as the segment's representative, W_p^e, and averaged over all the instances to compose the initial center, K_p , of the p-th cluster:

$$K_p = \frac{1}{E}\sum_{e=1}^{E} W_p^e ; \quad p = 1 \ldots (N * M) \tag{7}$$

Fig. 4 shows a plot of the first three coordinates of the initial centers, K_p, p = 1 … (N*M), N = 5, M = 3, for action class "ClimbLadder" in the MuHAVi action dataset [31]. Despite averaging over several instances of this action of varying length, the centers are well separated and show the dynamic progression of the observation values and their corresponding emitting states.

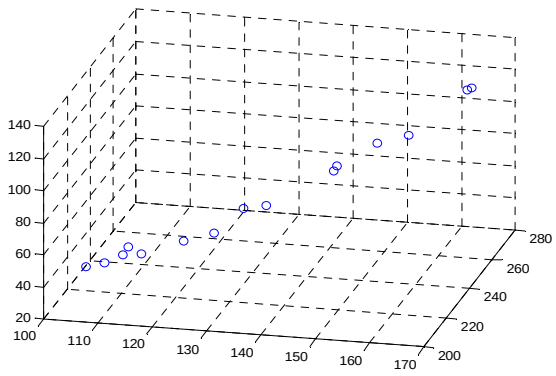

Fig. 4: The initial centers with the time-based initialization method (action class "ClimbLadder" from the MuHAVi action dataset [31]).

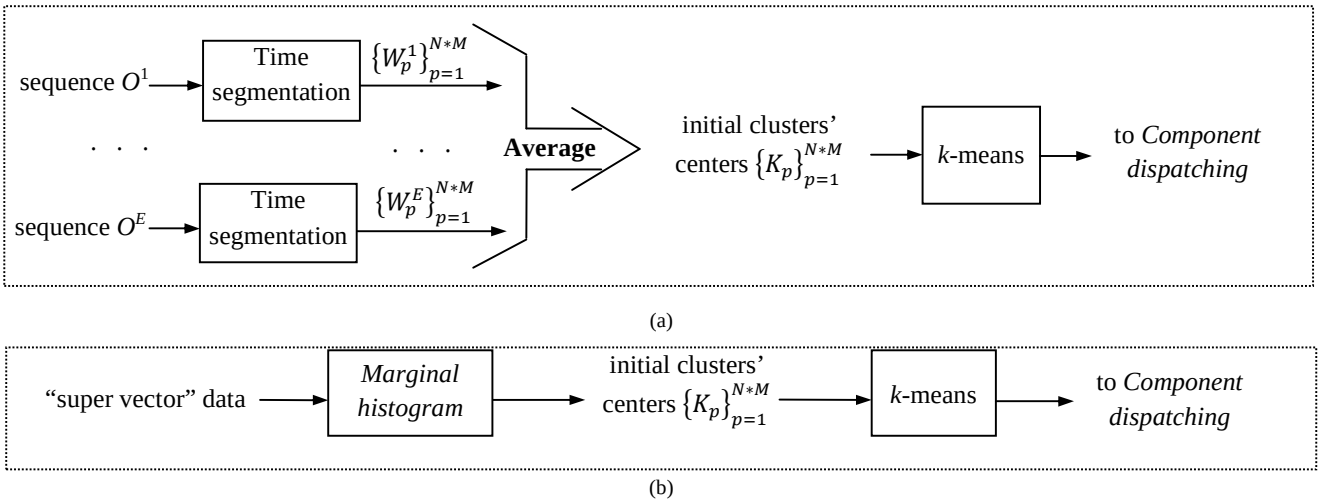

Fig. 5: a) the time-based initialization; b) the histogram-based initialization.

The time-based initialization is equivalent to assuming that the observations of each segment belong to the same state and that states occur in a "left-to-right" sequence with equal duration. In alternative to assuming equal duration for the segments, we could utilize a change-point detection technique to detect points of significant change in the observation sequence and segment it accordingly [32]. However, this would leave us with the hard problem of creating a correspondence between a variable number of segments in each training sequence and the fixed number of states in the HMM. Therefore, we choose to resort to these simplifying assumptions, leaving the responsibility to improve the mapping of observations to states to the following EM training.

2) Histogram-Based Initialization

An alternative to identifying the initial clusters' centers by time segmentation is that of ignoring the observations' time stamps and instead exploiting their proximity in feature space. We therefore turn to the marginal distribution of the observations, $p(o)$, and assume that each of its modes corresponds to a component of the observation probabilities, $b_i(o)$ (Fig. 5.b). While some interference between components from different states may occur, a reasonable expectation on modes' separation in high dimensions justifies this approach. We therefore propose an initialization method (named *marginal histogram of observations* hereafter) based on an approximated histogram of the data in the super vector. The aim of this method is to locate the positions of the main modes of $p(o)$ and use them as initial clusters' centers.

However, mode seeking in multi-dimensional data can prove inaccurate, especially when the training data are not sufficiently dense. To mollify this issue, we propose computing separate, 1-D histograms of each individual feature, and then constructing the initial centers *incrementally*, aggregating one feature at a time. This simplifying procedure is equivalent to assuming a convex shape for the clusters and independence between the features. This approach is similar in nature to that used by the FIRES clustering algorithm [33], with the main difference that the proposed algorithm is based on the distances between clusters' centers rather than on cluster overlap [33].

Assuming the feature set to be F-dimensional, for each feature f_i, i=1...F, the following four steps are performed (Fig. 6):

Step 1: A one-dimensional histogram is formed for feature f_i: its range is divided over (N*M) equally-sized bins and their counts computed.

Step 2: The bin with the highest count is selected as the first mode for feature f_i. Then, its count is decreased by 1/(N*M) of the total number of the observations. This subtraction may in some cases lead to a negative count for the bin.

Step 3: The procedure at the previous step is repeated (N*M) times. If a bin has a current negative count, it is excluded from the selection. Moreover, a single bin may end up being selected more than once if its count remains greater than that of all the other bins in successive iterations. At the end of these iterations, the selected bins account for the largest part of the observations and the whole procedure equates to a coarse quantization of the one-dimensional histogram. In addition, we consider the samples falling in each bin and we compute *the average of their first i coordinates*. Such i-dimensional means are noted as $\{C_p^{1:i}\}$, p = 1...(N*M), and represent the position of each bin in the space spanned by the first i coordinates.

While the above three steps construct the bins $\{C_p^{1:i}\}$ of a one-dimensional histogram, the next step joins such bins into a set of clusters' centers, $\{K_p\}$, whose dimensionality grows at every cycle and ends in the desired F-dimensional partition.

Step 4: For feature i = 1, the means $\{C_p^1\}$ are assigned to be the first coordinate of the final clusters' centers. At the i^{th} iteration, i = 2...F, the $\{K_p\}$ centers are already constructed up to their (i – 1) coordinate from the previous iterations and the $\{C_p^{1:i}\}$ vectors are currently computed. Thereafter, each $K_p^{1:i-1}$ is paired with vector $C_l^{1:i-1}$, l = 1...(N*M), with minimum

Euclidean distance from it:

$$l^* = argmin_{l=1\ldots(N*M)} \; ||(K_p^{1:i-1} - C_l^{1:i-1})||_{L2} \qquad (8)$$

Then, the value of the i-th coordinate, $C_{l^*}^i$, is assigned to be the i-th coordinate of K_p. The logic of this step is simple: to pair each partially-constructed K_p cluster with the closest $C_l^{1:i}$ bin based on the available coordinates. This approach is reminiscent of *incremental feature selection* techniques [34]; like any similar heuristic approach, it is reasonably fast, yet its outcome will depend on the arbitrary order in which the features appear in the feature set.

Fig. 7 depicts an example of the algorithm in the case of 2-dimensional features and $N*M$ = 6. The six histogram bins for each feature have been drawn alongside the two axes. The figure shows how the mode's mean of bin C_1^1 in the horizontal histogram (equal to 140) is used as the first coordinate of K_1. K_1 is then paired with bin $C_2^{1:2}$ in the vertical histogram since its first-coordinate mean, 136, is the closest to 140. Thus, the second coordinate of $C_2^{1:2}$ (equal to 231) is selected as the second coordinate of K_1.

C. Component Dispatching Methods

The ($N*M$) weighted Gaussian components obtained from the k-means step need to be later "dispatched" as modes of the observation probabilities of an HMM with N states, each with a mixture of M Gaussian components. This action may be regarded as not particularly critical since it may appear that changes to the modes' assignments will be compensated by corresponding changes to A, the state transition probabilities. However, there is a principled difference between data generated by components in the same mixture and data originated by components from different mixtures: the former are mutually independent conditioned on their state, while the latter depend sequentially. In addition, different combinations of components in the mixtures lead to different constructive interference between the modes, with changes to the overall shape of the distributions. As mentioned in subsection IV.A, the reference method for component dispatching is merely based on their appearance order in the set: certainly simple, yet completely arbitrary. Hence, in this subsection we propose two principled component dispatching methods and contrast them to the reference method.

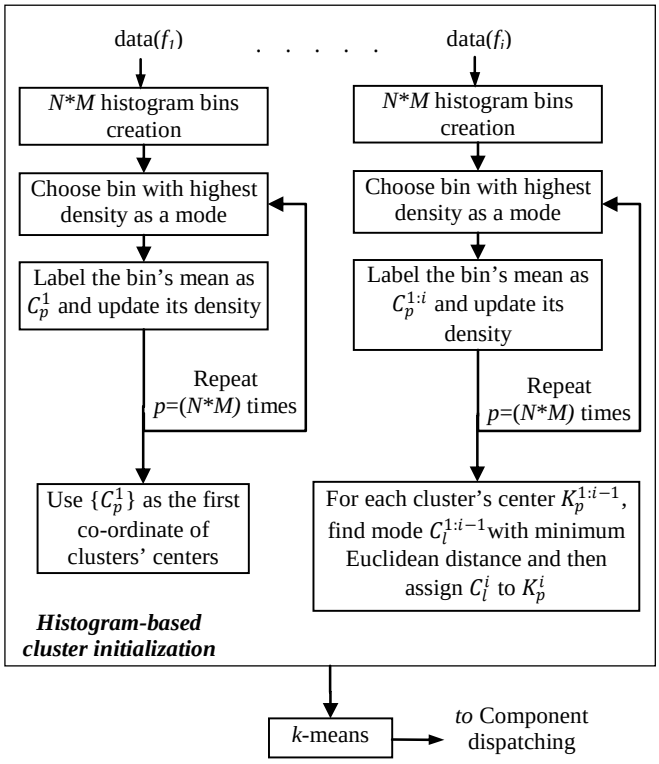

Fig. 6: Algorithm for the formation of the clusters' centers in the histogram-based initialization.

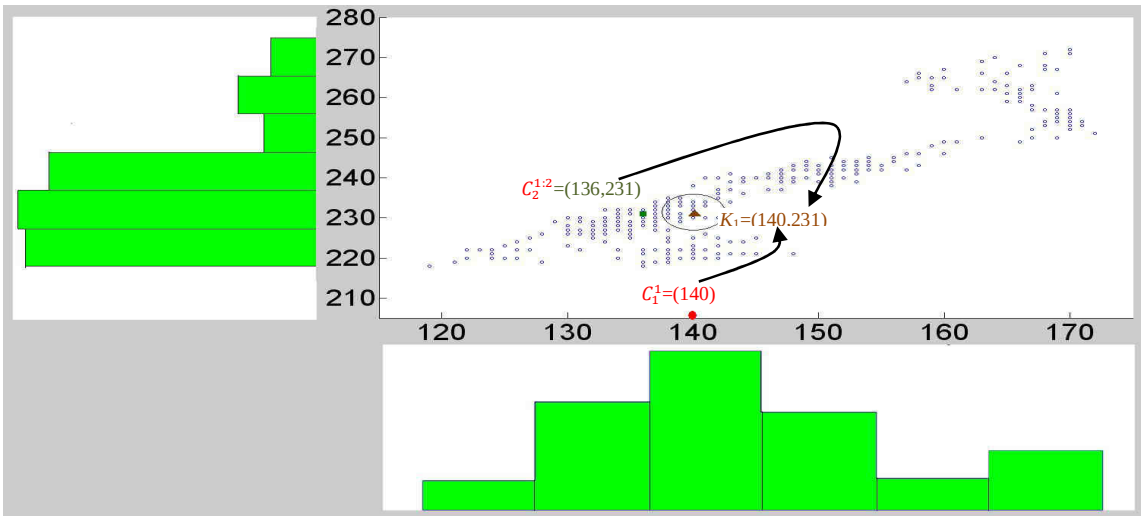

Fig. 7: An example of cluster's center calculation using the marginal histogram of observations method.

1) "Nearest Neighbors" Method

The main theme of this method is to put components with the closest centers in a single state. First, we compute all the Euclidean distances between pairs of components' centers. Then, we create all the possible partitions of components onto states and, for each partition, we accumulate the distance between all its component pairs. The partition with minimum total distance is selected as the best dispatching. The rationale of this dispatching is to attribute small measurement variations to the multiple modes of a single state, while attributing larger variations to switches between different states. The total number of possible partitions, T_P, for an N-state HMM with M components per state is:

$$T_p = \frac{\binom{M*N}{M} * \binom{M*(N-1)}{M} * \ldots * \binom{M}{M}}{N!} \quad (9)$$

where $\binom{n}{k} = \frac{n!}{k!*(n-k)!}$. As can be seen, T_P is, unfortunately, very high and the combinatorial exploration proves extremely time-consuming even for reasonably low values of N and M.

2) "Feature Sorting" Method

The goal of this method is to approximate the *nearest neighbors* dispatching with a much lower computational load. The (N*M) components' centers are F-dimensional vectors: therefore, they can be organized as an F x (N*M) matrix. Here, each row is first sorted in value order and the ranking of each cell in the sorted row retained. Then, the *average of the ranks* along each column is used to determine the overall rank of each component. Components are eventually dispatched to states in overall ranking order. This method enjoys a favorable O ((N*M)* log (N*M)) complexity.

V. Experiments

In the experiments, we have extensively evaluated the initialization methods by using three different human action video datasets - Weizmann [35], MuHAVi [31] and Hollywood [16] – and three diverse feature sets - the mask-based projection histograms [17, 36], the contour-based *sectorial extreme points* [37] and the popular STIP descriptors [21]. As software, we have used and extended Murphy's HMM toolbox for Matlab [29]. All the feature sets that we have used in the experiments are available for download at http://ieeexplore.ieee.org as companion material.

A. Feature Sets

As mentioned in Section II, human action recognition requires choosing a discriminative and workable feature set. If the videos permit an accurate selection of the actor' pixels (an operation often referred to as "mask extraction"), one can use computationally lightweight feature sets such as the projection histograms [17, 36] and the contour-based sectorial extreme points [37]. If such masks cannot be reliably extracted, local descriptors collected at *spatio-temporal interest points* (STIPs) provide an interesting, alternative feature set [21]. In the experiments, for two of the datasets - Weizmann and MuHAVi - we managed to exploit masks, while for Hollywood we had to rely on STIP descriptors.

1) The Projection Histograms Feature Set

The projection histograms features are computed by projecting each pixel of the actor' mask onto the image coordinate axes so as to form a horizontal and vertical histograms [17, 36]. As an action takes place, the two projection histograms reflect the changes in the object's shape and promise to prove action-discriminative. We have used histograms with 10 bins each and we have also added the centroid's coordinates to the feature vector to account for the actor's absolute position in the frame, for a total size F = 22. Fig. 8.b depicts the projection histograms for one frame of the Weizmann action video dataset (Fig. 8.a) [35].

2) The Sectorial Extreme Points Feature Set

The sectorial extreme points feature set is extracted by first computing the centroid of the actor's mask and then recording the coordinates of the mask pixels farthest from the centroid within five angular sectors in loose correspondence with anatomical parts [37]. We have also added the centroid's coordinates to the feature vector, for a total size F = 12. Fig. 8.c shows the extracted sectorial extreme points for the same frame of Fig. 8.a.

3) The STIP Descriptors Feature Set

STIPs are points of significant spatio-temporal change in a video (kind of "3D corners") which can be automatically detected by specific detection algorithms [21]. Upon detection of a STIP, a *descriptor* consisting of local space-time textural information is computed over a small volume centered on the point. In our experiments, we have used the STIP detector of Laptev *et al*. [21] which is based on an extension of the Harris detector to space-time. Fig.8.d shows the STIPs extracted from the same frame of Fig. 8.a. The descriptor is a 162-dimensional vector, obtained as the concatenation of a 72-element histogram of oriented gradient (HOG) and a 90-element histogram of optical flow (HOF).

B. Human Action Video Datasets

The *Weizmann* dataset includes 10 action classes of basic actions such as running, walking, jumping and others performed once or twice by 9 different subjects for a total of 93 video sequences [35]. Manually-extracted masks for all the videos are also available from the authors. This dataset is simple, yet it has been used in many previous papers and allows for the most extensive comparison with the literature.

The more recent *MuHAVi* dataset contains videos from 17 surveillance action classes such as climb a ladder, jump over a fence, look inside a car, draw graffiti and others. Each action is performed several times by 7 different subjects and captured by 8 cameras simultaneously [31]. To the best of our knowledge, this is the largest public action dataset to date in terms of combined number of action classes, subjects and cameras. For this paper, we have extracted the object masks automatically to obtain 398 action samples from camera four.

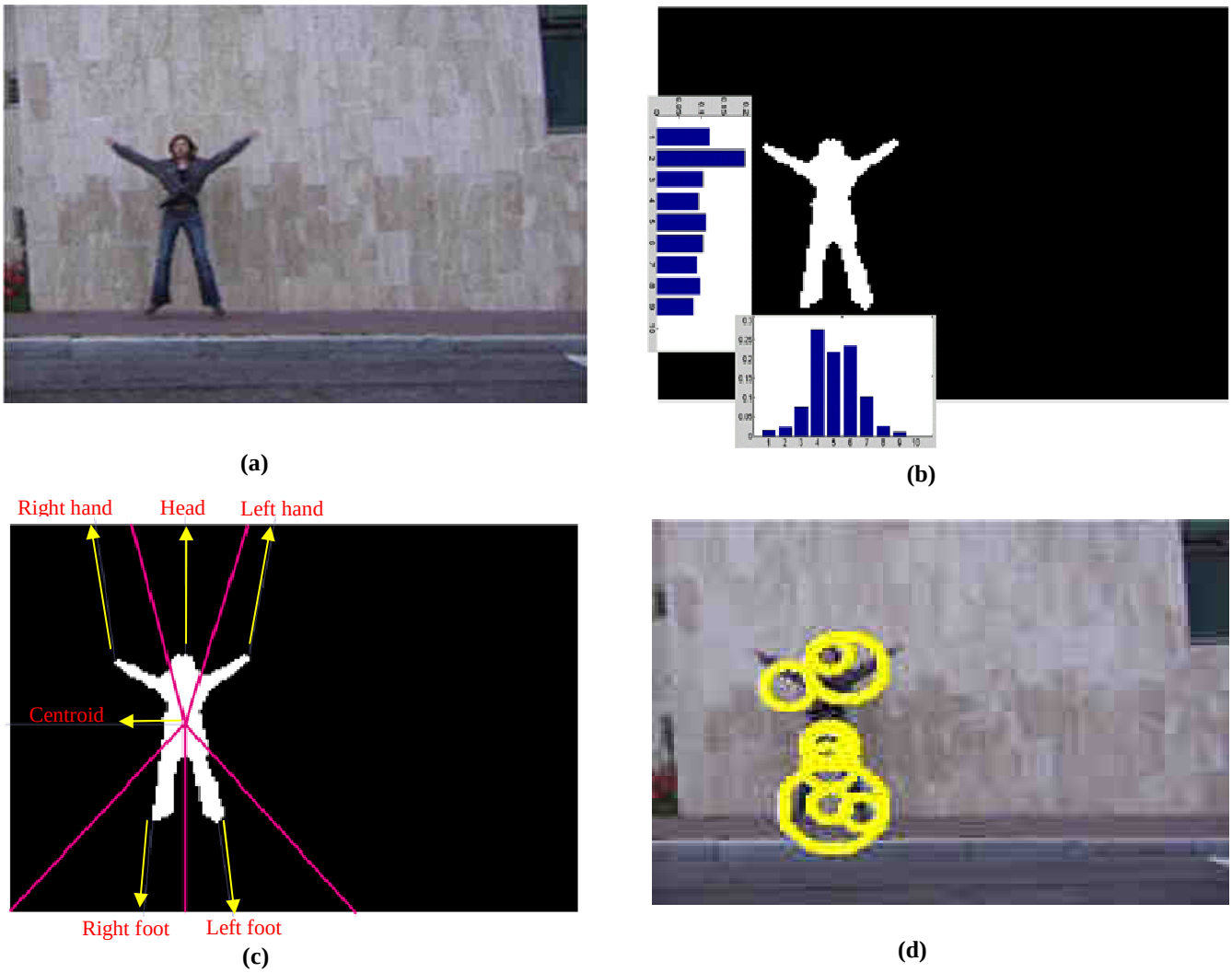

Fig. 8: Example of the feature sets: (a) a frame from the Weizmann dataset; (b) its projection histograms; (c) its sectorial extreme points; (d) its STIPs.

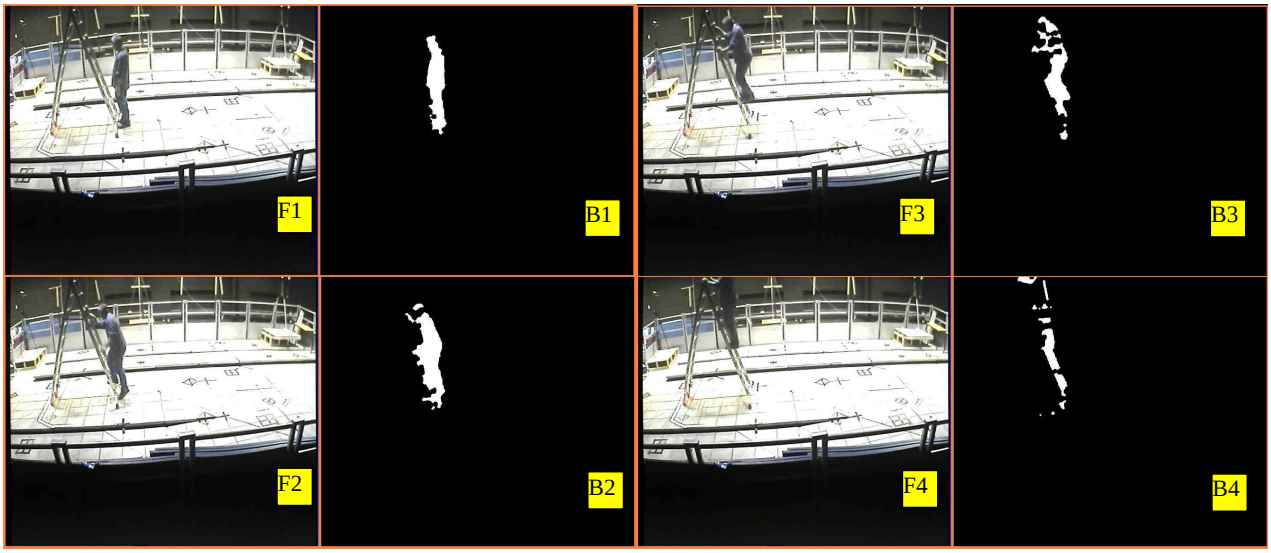

Fig. 9: Examples of frames and corresponding automated masks from the MuHAVi dataset.

The quality of these masks is lower and more realistic than that of the Weizmann masks. An example is shown in Fig. 9.

The *Hollywood* human action dataset is a very challenging action dataset of 8 action classes from commercial movies. In this type of video, actors are often shown only partially (torsos, faces, backs) and heavily occluded, with abrupt and frequent changes of view, camera movements and dynamic backgrounds [16]. The dataset consists of 475 video samples from 32 Hollywood movies with 8 action classes (answer phone, get out of car, handshake, hug, kiss, sit down, sit up and stand up). It comprises of a manually labeled training set (219 video samples for 231 action labels) and one manually labeled test set (211 video samples for 217 action labels), for a total of 448 action labels.

C. Experiments on the Cluster Initialization Methods

In the first set of experiments, we compared our two proposed cluster initialization methods, the *average of training instances* and the *marginal histogram of observations*, with the *random centers* cluster initialization method as reference. As component dispatching method, we applied the *appearance order* (subsection IV.A). For the *random centers*, we report the average accuracy and standard deviation over six different random starts, and also the best accuracy out of the six starts. The number of starts was chosen as the highest practicable number for the experiments, where each training session lasted in the order of sixteen hours and tens of sessions had to be run for the various combinations of the parameters.

As validation approach over Weizmann and MuHAVi datasets, we have used the "leave-one-subject-out" cross validation method; i.e. in each run we leave one subject out during training and we use it for testing. This validation procedure is realistic since in real applications subjects would not have been seen during training. The final accuracy result is the average over the various subjects, with 9 folds for Weizmann and 7 folds for MuHAVi. For the Hollywood dataset, we have used the dictated training and test sets for comparability.

Since the number of HMM states, N, and the number of components per state, M, are hyperparameters in the Baum-Welch algorithm and cannot be determined by maximum likelihood estimation, we simply adopt exhaustive search over a plausible range, N, $M \in \{1\ldots6\}$, and choose the best combination based on cross-validation accuracy.

1) Experiments over the Weizmann Dataset

Tables I and II report the classification accuracy over the Weizmann dataset by using the sectorial extreme points and projection histograms feature sets, respectively. In the tests with the sectorial extreme points feature set, the *average of training instances* and the *marginal histogram of observations* obtained the highest accuracies, 96.8% and 95.7%, respectively, higher than the average of the six runs of *random centers*, 94.6% and comparable to their best, 96.8%. Similar results were achieved with the projection histograms feature set, with 87.1% and 88.2% as the highest accuracies of *average of training instances* and *marginal histogram of observations*, respectively, compared to 85.8% and 88.2% as the average and best of the 6 runs of *random centers*. This proves that one single training sessions with the methods proposed in this paper can be equally effective as multiple training sessions with random trials. If applied an unbounded number of times, it is almost certain that the random initializations would eventually provide the highest accuracy

TABLE I
CLASSIFICATION ACCURACY (%) FOR THE VARIOUS CLUSTER INITIALIZATION METHODS WITH THE SECTORIAL EXTREME POINTS FEATURE SET AND THE WEIZMANN DATASET.

***Random centers* (average and std. dev. of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	**94.6±0.0**	91.2±1.1	92.5±0.0	93.5±0.7	93.9±0.9	93.9±1.3
N=2	94.1±0.9	94.1±1.8	93.5±1.5	90.7±2.0	91.8±2.9	90.9±1.5
N=3	93.5±2.2	92.3±2.3	92.8±1.1	89.8±1.5	90.0±2.3	90.7±1.5
N=4	93.7±0.8	91.2±2.6	89.4±2.0	88.0±2.4	89.1±2.5	88.4±1.9
N=5	92.7±1.3	90.7±1.8	90.7±1.9	90.0±2.0	87.6±1.3	87.8±2.5
N=6	92.1±1.3	89.6±2.5	88.5±1.9	88.7±2.0	88.2±2.8	87.6±2.2

***Random centers* (best of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	94.6	92.5	92.5	94.6	94.6	94.6
N=2	95.7	**96.8**	95.7	93.5	93.5	92.5
N=3	94.6	**96.8**	94.6	91.4	93.5	92.5
N=4	94.6	92.5	91.4	91.4	92.5	91.4
N=5	93.5	92.5	93.5	93.5	89.2	90.3
N=6	94.6	91.4	91.4	91.4	92.5	91.4

Average of training instances

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	94.6	91.4	92.5	89.2	**96.8**	93.5
N=2	93.5	92.5	93.5	90.3	95.7	91.4
N=3	93.5	92.5	88.2	89.2	91.4	86.0
N=4	93.5	91.4	90.3	89.2	91.4	89.2
N=5	91.4	91.4	90.3	91.4	89.2	79.6
N=6	94.6	89.2	88.2	90.3	78.5	76.3

Marginal histogram of observations

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	94.6	91.4	94.6	91.4	**95.7**	**95.7**
N=2	93.5	93.5	94.6	90.3	90.3	89.2
N=3	93.5	93.5	91.4	87.1	91.4	91.4
N=4	92.5	92.5	88.2	87.1	88.2	87.1
N=5	93.5	90.3	88.2	91.4	90.3	89.2
N=6	91.4	88.2	89.2	87.1	87.1	86.0

for any combination of *N*, *M*. However, such a brute force approach is just impractical. While the best accuracy of the *average of training instances* is higher than that of the *marginal histogram of observations* in the test with the first feature set (sectorial extreme points), the situation is reversed when using the other (projection histograms). However, the differences in accuracy are small and either method can be vetted as a suitable one-off, initialization solution.

TABLE II
CLASSIFICATION ACCURACY (%) FOR THE VARIOUS CLUSTER INITIALIZATION METHODS WITH THE PROJECTION HISTOGRAMS FEATURE SET AND THE WEIZMANN DATASET.

***Random centers* (average and std. dev. of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	75.3±0.0	72.0±0.0	72.0±0.0	72.0±0.0	72.0±0.0	72.0±0.0
N=2	83.7±1.1	80.8±2.7	81.0±2.2	80.6±1.0	80.8±1.3	81.4±1.1
N=3	82.8±0.7	81.0±0.9	80.5±1.9	82.1±1.9	84.1±1.7	81.0±2.2
N=4	83.2±2.1	84.4±1.1	83.0±1.9	82.8±2.3	83.9±1.4	83.5±2.2
N=5	83.9±1.0	85.1±1.4	83.7±3.6	84.9±1.7	84.2±0.6	83.7±2.5
N=6	85.3±1.1	84.4±0.9	**85.8±2.0**	**85.8±1.9**	85.7±2.4	83.7±1.6

***Random centers* (best of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	75.3	72.0	72.0	72.0	72.0	72.0
N=2	84.9	84.9	83.9	81.7	82.8	82.8
N=3	83.9	81.7	83.9	84.9	87.1	83.9
N=4	84.9	86	84.9	84.9	86.0	87.1
N=5	84.9	87.1	**88.2**	86.0	84.9	86.0
N=6	87.1	84.9	87.1	**88.2**	**88.2**	86.0

Average of training instances

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	75.3	73.1	72.0	72.0	72.0	72.0
N=2	84.9	81.7	76.3	81.7	79.6	80.6
N=3	79.6	83.9	81.7	83.9	81.7	**87.1**
N=4	82.8	84.9	82.8	83.9	82.8	80.6
N=5	84.9	86.0	81.7	**87.1**	84.9	79.6
N=6	84.9	86.0	86.0	80.6	82.8	80.6

Marginal histogram of observations

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	75.3	72.0	72.0	72.0	72.0	72.0
N=2	83.9	80.6	84.9	80.6	81.7	80.6
N=3	81.7	81.7	80.6	83.9	80.6	79.6
N=4	81.7	83.9	81.7	79.6	84.9	83.9
N=5	83.9	87.1	83.9	81.7	83.9	**88.2**
N=6	87.1	84.9	84.9	83.9	86.0	**88.2**

TABLE III
CLASSIFICATION ACCURACY (%) FOR THE VARIOUS CLUSTER INITIALIZATION METHODS WITH THE SECTORIAL EXTREME POINTS FEATURE SET AND THE MUHAVI DATASET.

***Random centers* (average and std. dev. of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9±0.0	88.2±0.5	90.7±0.4	91.4±0.5	91.4±0.3	91.8±0.7
N=2	88.5±0.5	91.7±0.4	91.9±0.4	92.2±0.6	91.8±0.8	91.6±0.9
N=3	91.5±0.5	92.0±0.6	**92.4±0.6**	92.1±1.3	91.8±0.5	91.8±0.7
N=4	91.9±0.7	92.1±0.6	92.1±0.5	91.8±0.9	91.3±0.6	91.5±0.4
N=5	92.2±1.1	92.3±0.3	92.2±0.9	91.7±0.7	91.3±1.1	90.6±0.8
N=6	92.3±0.9	92.0±0.6	91.9±0.9	91.4±0.8	90.2±0.4	90.4±1.3

***Random centers* (best of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9	88.9	91.2	92.2	91.7	92.7
N=2	89.4	92.2	92.5	93.2	93.0	92.7
N=3	92.2	93.2	93.2	**94.0**	92.5	92.5
N=4	92.7	92.7	92.7	92.7	92.0	92.2
N=5	93.7	92.7	93.2	92.7	92.5	91.5
N=6	93.2	92.7	93.2	92.5	90.7	92.2

Average of training instances

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9	88.9	90.5	90.7	92.5	**93.7**
N=2	89.4	91.5	92.2	91.7	90.7	91.7
N=3	91.2	**93.7**	91.7	93.0	92.5	92.2
N=4	91.7	91.7	92.5	93.2	90.2	92.2
N=5	92.5	91.2	92.0	91.7	91.7	91.2
N=6	93.0	93.5	91.7	91.2	91.2	89.7

Marginal histogram of observations

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9	87.7	89.7	91.7	91.0	91.7
N=2	88.4	92.0	91.7	92.5	92.2	92.2
N=3	91.5	93.0	91.7	91.5	91.7	91.7
N=4	92.7	92.0	91.0	92.5	90.5	89.7
N=5	91.0	**93.2**	92.0	92.7	91.7	90.7
N=6	92.5	91.7	92.2	89.9	90.7	89.7

2) Experiments over the MuHAVi Dataset

The classification accuracy results with the various cluster initialization methods over the MuHAVi dataset are shown in Tables III and IV. Again, the results show that the highest accuracies achieved with the *average of training instances* and the *marginal histogram of observations* are higher than the average of the six runs with the *random centers* and comparable with their best. With the projection histograms feature set, the maximum accuracy of the *average of training instances* method (92.0%) proved even higher than the best of six random starts (91.7%).

TABLE IV
CLASSIFICATION ACCURACY (%) FOR THE VARIOUS CLUSTER INITIALIZATION METHODS WITH THE PROJECTION HISTOGRAMS FEATURE SET AND THE MUHAVI DATASET.

***Random centers* (average and std. dev. of 6 runs)**

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	83.2±0.0	88.0±0.3	88.1±0.1	88.8±0.3	89.0±0.5	89.0±0.3
N=2	87.3±0.3	88.1±0.9	89.7±0.6	89.4±0.7	89.1±0.4	89.5±0.6
N=3	87.3±0.4	89.3±0.7	89.8±1.0	89.4±0.8	**90.2±0.6**	89.7±0.6
N=4	87.7±0.4	89.2±0.6	89.3±0.5	89.4±0.6	88.8±0.7	88.9±0.7
N=5	88.0±0.9	88.5±0.5	89.3±1.2	89.6±0.9	89.0±0.6	89.6±0.5
N=6	88.6±0.4	88.7±0.8	89.2±1.2	89.1±0.7	89.4±0.8	88.9±0.2

***Random centers* (best of 6 runs)**

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	83.2	88.4	88.2	89.2	89.7	89.4
N=2	87.7	89.4	90.5	90.2	89.7	90.2
N=3	87.7	90.2	**91.7**	90.5	91.2	90.7
N=4	88.2	89.9	89.9	90.2	89.4	89.7
N=5	89.4	89.2	**91.7**	91.0	89.7	90.5
N=6	89.2	89.7	90.2	90.2	90.5	89.2

Average of training instances

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	83.2	87.9	88.4	88.7	88.9	88.7
N=2	87.2	87.9	89.2	88.7	87.9	89.9
N=3	87.2	90.2	89.2	89.7	**92.0**	89.7
N=4	88.4	88.7	88.2	89.2	91.0	89.4
N=5	88.9	89.2	89.4	90.2	88.7	88.7
N=6	88.7	88.7	91.0	89.4	89.9	89.7

Marginal histogram of observations

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	83.2	87.9	87.7	88.4	89.2	88.7
N=2	86.9	89.9	90.7	88.4	88.7	**91.5**
N=3	86.9	88.7	89.9	90.7	89.7	88.7
N=4	88.4	89.9	90.2	90.5	89.4	90.5
N=5	89.2	89.4	87.9	89.2	88.2	87.9
N=6	87.4	88.9	89.4	88.7	89.7	88.9

TABLE V
CLASSIFICATION ACCURACY (%) FOR THE VARIOUS CLUSTER INITIALIZATION METHODS WITH THE STIP FEATURE SET AND THE HOLLYWOOD DATASET.

***Random centers* (average and std. dev. of 6 runs)**

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	17.1±0.0	**26.8±1.6**	26.0±1.6	25.0±0.6	26.0±0.7	26.0±2.0
N=2	17.1±1.7	26.2±0.9	25.8±1.1	25.8±0.8	25.3±0.7	25.6±0.9
N=3	17.1±0.7	25.9±1.1	25.0±1.2	25.6±0.7	25.4±1.6	24.7±1.1
N=4	18.7±1.4	25.9±0.3	25.4±1.0	25.0±0.6	24.7±1.1	24.4±0.9
N=5	21.7±2.9	25.7±1.3	25.5±0.6	24.5±0.9	24.8±1.0	24.0±0.7
N=6	21.4±1.9	25.6±1.1	24.4±1.6	24.7±1.0	24.7±0.6	23.7±1.2

***Random centers* (best of 6 runs)**

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	17.1	28.6	28.6	25.8	26.7	**29.9**
N=2	19.4	26.7	26.7	26.7	26.3	26.7
N=3	18.0	27.6	26.7	26.3	27.2	25.8
N=4	20.3	26.3	27.2	25.8	26.3	25.3
N=5	26.3	27.6	26.3	25.8	26.7	24.9
N=6	24.4	27.2	26.7	25.8	25.3	25.8

Average of training instances

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	17.1	**26.7**	24.9	**26.7**	26.3	24.9
N=2	15.7	26.3	25.3	25.3	16.1	14.3
N=3	13.8	25.3	14.7	12.4	14.3	17.5
N=4	18.9	25.3	13.4	16.6	15.7	7.4
N=5	20.7	17.5	12.4	16.6	8.8	10.6
N=6	20.7	14.3	18.0	7.4	10.6	10.6

Marginal histogram of observations

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	17.1	25.8	25.8	25.8	27.6	**28.1**
N=2	19.4	25.3	26.3	25.8	24.4	25.8
N=3	18.9	27.2	25.8	25.8	24.9	24.9
N=4	16.6	25.3	25.8	26.3	24.4	25.3
N=5	19.8	22.6	24.9	24.0	24.4	26.7
N=6	22.1	25.3	24.9	23.5	25.3	24.9

3) Experiments over the Hollywood Dataset

The Hollywood human action dataset is one of the most challenging action datasets released to date. As mentioned in Section V.A, it is virtually impossible to extract sufficiently accurate actor masks from Hollywood and we have therefore resorted to STIP descriptors. The number of detected STIPs for each video frame may be zero, one or more than one. For frames with multiple STIPs, we have chosen to adopt the method from [38] to combine and average all the extracted descriptors into one feature vector. For frames with no STIPs, no feature vector is generated.

Table V reports the classification accuracies with the various cluster initialization methods. The achieved results are in line with the results from the other datasets (Weizmann and MuHAVi). The maximum achieved accuracy with the *average of training instances* and the *marginal histogram of observations* methods are 26.7% and 28.1%, respectively, compared to 26.8% and 29.9% as the average and the highest achieved accuracy over 6 runs with the *random centers* method. For this dataset, the result from the *marginal histogram of observations* method is more noticeable. Chance accuracy (the accuracy of assigning a class by a uniformly random decision) for this dataset is approximately 12% and results in the current range are in line with the state of the art as will be shown in subsection V.F.

4) Divergence Analysis

Given that the time-based and histogram-based initialization methods lead to different initializations and eventual accuracy outcomes, in this paper we also attempt to identify a heuristic method to choose between them straight after the determination of the initial parameters, prior to performing HMM training and cross-validation.

To this aim, we have assumed that a satisfactory initialization should provide us with *a set of Gaussian components which are as separated as possible*. Separation between components is likely in feature spaces of such dimensionality and leads to more reliable mode identification. We have therefore adopted a divergence measure for measuring the separation between all possible pairs of Gaussian components and select the initialization method producing the smallest overlap between components. A common divergence measure between two density functions is the Kullback-Leibler divergence [39], which is defined as follows:

$$D_{KL}(p||q) = \int_{-\infty}^{+\infty} p(x)\, ln\left(\frac{p(x)}{q(x)}\right) dx \tag{10}$$

where p and q are two distributions of a continuous random variable, x. For two multivariate F-dimensional Gaussian densities, $\mathcal{N}(x|\mu_p, \Sigma_p)$ and $\mathcal{N}(x|\mu_q, \Sigma_q)$, this divergence has a well-known closed-form solution [40]:

$$D_{KL}(\mathcal{N}_p||\mathcal{N}_q) = \frac{1}{2}\left\{ ln\left[\frac{det(\Sigma_q)}{det(\Sigma_p)}\right] + tr\left({\Sigma_q}^{-1}\Sigma_p\right) + \left(\mu_p - \mu_q\right)^T {\Sigma_q}^{-1}\left(\mu_p - \mu_q\right) - F \right\} \tag{11}$$

However, a weighted Gaussian component, noted as $c_l\mathcal{N}(x|\mu_l, \Sigma_l)$, is a denormalized density in that its total probability adds up to its weight, c_l, with $c_l \leq 1$. Therefore, we make use of the *extended Kullback-Leibler divergence* as defined in [41] which can be applied to denormalized density functions:

$$D_{eKL}(p||q) = \int_{-\infty}^{+\infty} \left[q(x) - p(x) + p(x)\, ln\left(\frac{p(x)}{q(x)}\right)\right] dx \tag{12}$$

The extended Kullback-Leibler divergence for two weighted Gaussian components, $p(x) = c_p\mathcal{N}(x|\mu_p, \Sigma_p)$ and $q(x) = c_q\mathcal{N}(x|\mu_q, \Sigma_q)$, is therefore derived here as follows:

$$\begin{aligned} &D_{eKL}(p||q) = \\ &= \int_{-\infty}^{+\infty} \left[c_q\mathcal{N}\left(x\middle|\mu_q, \Sigma_q\right) - c_p\mathcal{N}\left(x\middle|\mu_p, \Sigma_p\right) \right. \\ &\left. + c_p\mathcal{N}\left(x\middle|\mu_p, \Sigma_p\right) ln\left(\frac{c_p\mathcal{N}\left(x\middle|\mu_p, \Sigma_p\right)}{c_q\mathcal{N}\left(x\middle|\mu_q, \Sigma_q\right)}\right)\right] dx = \\ &= c_q - c_p + c_p\left[ln\left(\frac{c_p}{c_q}\right) + D_{KL}(\mathcal{N}_p||\mathcal{N}_q)\right] \end{aligned} \tag{13}$$

We have used (11) to compute the pair-wise divergence between all components after cluster initialization. We have then defined an arbitrary threshold, T_{KL}, (set to 1 in the experiments) and counted the number of pairs for which divergence was below such a threshold. These pairs are regarded as "undesirable", and the initialization method providing their smallest number is selected.

This selection approach was tested over the four parameter combinations with the highest accuracies from Tables III and IV. Results are shown in Table VI: the selection approach offers a successful prediction if the higher of the two accuracies (in boldface) is achieved in correspondence with the smallest number of undesirable pairs (also in boldface). As can be seen, in three cases out of

four the divergence criterion provided us with a correct prediction of the more accurate of the two initialization methods (a lower number of undesirable pairs led to higher accuracy). However, the reader ought to keep in mind the heuristic nature of this criterion and that it can only be seen as a precursor of the cross-validation accuracy.

D. Experiments on the Component Dispatching Methods

The second set of experiments was designed to explore the best component dispatching method among those described in subsection IV.C. As explained before, the computational complexity of the *nearest neighbors* dispatching method is unmanageable. Therefore, we decided to run the experiment only between *feature sorting* and the baseline method *appearance order*. The experiments were cross-run with all the cluster initialization methods: *random centers*, *average of training instances* and *marginal histogram of observations*. The achieved accuracies over the MuHAVi dataset using the *feature sorting* dispatching method and the *sectorial extreme points* feature set are shown in Table VII. For ease of comparison, in the left column we have reported the accuracies of the *appearance order* dispatching method from Table III.

The results in Table VII show that using *feature sorting* for dispatching the components to states improves the highest accuracy of the *marginal histogram of observations* initialization method by 1.3% (from 93.2% to 94.5%) and the best accuracy of the *random centers* initialization method by 0.7% (from 94.0% to 94.7%). No improvement was instead achieved for *average of training instances*. Even more notably, the combination of a one-off cluster initialization (*marginal histogram of observations)* with an "intelligent" dispatching (*feature sorting*) achieved 94.5% accuracy which is higher than the highest accuracy of the six random starts (94.0%).

TABLE VI
DIVERGENCE ANALYSIS FOR THE TWO ONE-OFF INITIALIZATION METHODS.

Feature set	Total pairs	*Average of training instances*		*Marginal histogram of observations*	
		Accuracy	Undesirable pairs	Accuracy	Undesirable pairs
sectorial extreme points	3570	**93.7%**	**137**	91.7%	141
	10710	91.2%	1319	**93.2%**	**1215**
projection histograms	15708	89.9%	1346	**91.5%**	**1031**
	24990	**92.0%**	2548	89.7%	**1985**

TABLE VII
ACCURACY COMPARISON (%) BETWEEN COMPONENT DISPATCHING METHODS:
APPEARANCE ORDER (LEFT COLUMN); FEATURE SORTING (RIGHT COLUMN), FOR VARIOUS CLUSTER INITIALIZATION METHODS.

Appearance order

***Random centers* (best of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9	88.9	91.2	92.2	91.7	92.7
N=2	89.4	92.2	92.5	93.2	93.0	92.7
N=3	92.2	93.2	93.2	**94.0**	92.5	92.5
N=4	92.7	92.7	92.7	92.7	92.0	92.2
N=5	93.7	92.7	93.2	92.7	92.5	91.5
N=6	93.2	92.7	93.2	92.5	90.7	92.2

Average of training instances

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9	88.9	90.5	90.7	92.5	**93.7**
N=2	89.4	91.5	92.2	91.7	90.7	91.7
N=3	91.2	**93.7**	91.7	93.0	92.5	92.2
N=4	91.7	91.7	92.5	93.2	90.2	92.2
N=5	92.5	91.2	92.0	91.7	91.7	91.2
N=6	93.0	93.5	91.7	91.2	91.2	89.7

Feature sorting

***Random centers* (best of 6 runs)**

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9	88.2	91.2	92.2	92.5	92.2
N=2	88.9	92.5	93.5	92.7	94.0	93.0
N=3	91.5	92.7	92.5	92.2	92.5	92.5
N=4	93.2	92.7	**94.7**	92.7	92.7	91.5
N=5	92.7	93.7	93.0	91.7	92.0	91.7
N=6	93.2	92.7	92.0	92.5	90.2	90.5

Average of training instances

	M=1	**M=2**	**M=3**	**M=4**	**M=5**	**M=6**
N=1	84.9	88.9	90.5	90.7	92.5	**93.7**
N=2	89.4	91.5	92.5	91.5	90.5	92.7
N=3	91.2	92.7	92	92.5	92.2	91.2
N=4	91.7	92.7	93.5	91.7	91.7	91.2
N=5	92.5	92	92	90.5	90.5	91.2
N=6	93	92.7	91.5	91.5	91.5	90.7

Marginal histogram of observations (continued)

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	84.9	87.7	89.7	91.7	91.0	91.7
N=2	88.4	92.0	91.7	92.5	92.2	92.2
N=3	91.5	93.0	91.7	91.5	91.7	91.7
N=4	92.7	92.0	91.0	92.5	90.5	89.7
N=5	91.0	**93.2**	92.0	92.7	91.7	90.7
N=6	92.5	91.7	92.2	89.9	90.7	89.7

	M=1	M=2	M=3	M=4	M=5	M=6
N=1	84.9	87.7	89.7	91.7	91	91.7
N=2	88.4	91.7	92	91.2	91.7	92
N=3	91.5	**94.5**	92.5	93.2	91.7	92.2
N=4	92.7	92	91.7	91.7	91	90.2
N=5	91	91.2	91.5	91.7	91.2	91.5
N=6	92.5	91.7	92.2	89.9	90.7	89.7

E. Computational time analysis

Given the interesting accuracy reported by the proposed methods, we have proceeded to measure their execution time to assess if they introduce any remarkable computational overhead compared to the conventional initialization. Table VIII reports a break-down of the average execution times over repeated runs for the various algorithms. We have chosen to run a small-size training session to be able to include the *nearest-neighbors dispatching* method in the comparison. With any larger size of the training session (in terms of number of samples, states, or components), the execution time of *nearest-neighbors dispatching* becomes unmanageable. Table VIII shows that the execution times of all initialization components are negligible with respect to an EM run, with a maximum of 2.9% for the *marginal histograms* clustering. Therefore, using the proposed initializations in place of multiple EM runs saves an amount of training time directly proportional to their number, without incurring any noticeable overhead. As Table VIII shows, the execution time of *Nearest neighbors dispatching* is by far larger than that of EM itself (828.2%) and therefore impractical in any scenario.

TABLE VIII
EXECUTION TIMES (IN SECONDS) FOR THE VARIOUS INITIALIZATION COMPONENTS IN COMPARISON WITH EM.

Algorithm	Execution time (s)	Percentage vs. EM
Random centres	0.14	0.0
Time-based segmentation	1.22	0.1
Marginal histograms	48.6	2.9
Appearance order	0.005	0.0
Feature sorting	0.3	0.0
Nearest neighbors	14,162	828.2
EM training	1,710	(100.0)

F. Comparison with Other Recognition Approaches

HMM has been utilized for many years as an effective sequential classifier not only for action recognition, but also in domains as diverse as speech recognition and genomics. However, many alternative approaches have reported high accuracy, with *Bag-of-Features (BoF)* certainly the most prominent [15]. BoF was originally inspired by the Bag-of-Words model for document classification which computes a histogram of words for each document and uses it as a measurement for its classification. BoF uses a similar idea for the classification of images and videos, yet using local descriptors (such as the STIP descriptors presented in subsection V.A) in place of words. However, local descriptors belong to vector spaces and require a quantization step before histograms can be computed. To this aim, a training stage is run by extracting all the descriptors of a training video set and forming *D* bins by a *k*-means clustering algorithm. At run time, for each video, all the descriptors are extracted and associated to bins, their histogram computed, and eventually used for classification of the video by using any well-known classifier such as the support vector machine (SVM), nearest neighbors, AdaBoost or others [15].

In our experiments, the number of clusters, *D*, was selected over a logarithmic range, {16, 32, 64, 128, 256}. Since the *k*-means algorithm generates different results at each run, we also repeated the cluster formation 6 times. For classification, we have used SVM and AdaBoost. For SVM, we have used the LIBSVM library [42] with two different kernels: the linear kernel and the radial basis function, with a grid-search over parameter *nu* in {0.05÷ 0.35, step 0.05}. For AdaBoost, we have used BFTree as weak classifier. Table IX compares the best accuracy achieved over the MuHAVi dataset (with sectorial extreme points) and Hollywood (using the STIP descriptors).

Table IX shows that the accuracy achieved by HMM and BoF over these two datasets is generally comparable. HMM achieved a higher accuracy over the MuHAVi dataset (94.0%), while HMM and BoF achieved the same best accuracy over Hollywood (29.9%). In all experiments with BoF, SVM outperformed AdaBoost, with increases ranging between 3.3% and 6.8%. Again, in all experiments with BoF the accuracy depended significantly on the outcome of the clustering step, with significant differences between the best accuracy and the average accuracy over the 6 trials (from 1.6% to 5.6%). The best results for SVM were achieved with the RBF kernel for MuHAVi and with the linear kernel for Hollywood.

TABLE IX
COMPARISON OF CLASSIFICATION ACCURACY (%) FOR VARIOUS CLASSIFIERS OVER THE MUHAVI AND HOLLYWOOD DATASETS.

Dataset	Feature set	Accuracy	HMM			BoF/ SVM	BoF/ AdaBoost
			Reference method	*Average of training instances*	*Marginal histogram of observations*		
MuHAVi	Sectorial extreme points	Best	94.0%	93.7%	93.2%	86.7%	79.9%
		Average	92.1%			81.7%	78.3%
Hollywood	STIP descriptors	Best	29.9%	26.7%	28.1%	29.9%	24.9%
		Average	26.0%			24.3%	21.0%

VI. CONCLUSION

In this paper, we have reconsidered the long-standing problem of parameter initialization in model training. Given that the objective functions are affected by multiple maxima, varying the initialization has a substantial impact on the learned model and, in turn, cross-validation performance. In the paper, we have proposed two one-off initialization methods for the training of HMM in contrast to the usual approach of repeated, random initializations. The two methods are based on a different rationale: in the first (*sequentiality*), each training sequence is divided into *N* segments of equal length. The mean value of each segment is computed and averaged over corresponding segments across the training sequences. Such averages are then used as the initial centers. The underlying assumption is a linear progression of the human action, leaving the discovery of non-linearities to the subsequent HMM training phase. The second approach (*proximity in feature space*) is based on the marginal histogram of the observations and makes use of an incrementally-built histogram to mollify the issues associated with histograms in high-dimensional spaces. Either approach is followed by a cluster dispatching step based on a heuristic feature sorting procedure. In addition, an empirical measure of divergence has been proposed as an indicator of the quality of the initialization.

The proposed one-off approaches achieve high recognition accuracy from a single model training, thus saving substantial learning time compared to multiple-starts methods. Experimental results over three human action recognition datasets have showed that the proposed initializations are capable of achieving better accuracy than the average of multiple (six) random initializations and comparable to their best. In our experiments, reducing the training sessions from six to one has permitted a reduction in training time of over 83%, from approximately four days down to sixteen hours (this figure varies depending on the configurations used in the experiment). We also argue that the proposed approaches are general and can be usefully applied to HMM classification in other domains including genomics, speech recognition, network traffic categorization and others. In addition, the approach can be extended to more complex, discrete latent-state models such as switching HMMs and more general dynamic Bayesian networks which also suffer from an equivalent problem of local maxima and resort to comparable initialization heuristics. Such extensions are the scope of our current work.

REFERENCES

[1] R. Poppe, "A Survey on Vision-Based Human Action Recognition," *Image and Vision Computing (IVC)*, vol. 28, pp. 976-990, Jun. 2010.

[2] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine Recognition of Human Activities: A Survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp. 1473-1488, Nov. 2008.

[3] T. B. Moeslund, A. Hilton, and V. Krüger, "A Survey of Advances in Vision-Based Human Motion Capture and Analysis," *Computer Vision and Image Understanding (CVIU)*, vol. 104, pp. 90-126, Nov. - Dec. 2006.

[4] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, pp. 257-286, Feb. 1989.

[5] Q. Shi, L. Wang, L. Cheng, and A. Smola, "Discriminative Human Action Segmentation and Recognition using Semi-Markov Model," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska, 2008, pp. 1-8.

[6] Y.-C. Wu, H.-S. Chen, W.-J. Tsai, S.-Y. Lee, and J.-Y. Yu, "Human Action Recognition Based on Layered-HMM," in *IEEE Intl. Conf. on Multimedia and Expo (ICME)*, Hannover, Germany, 2008, pp. 1453-1456.

[7] L. Wang and D. Suter, "Recognizing Human Activities from Silhouettes: Motion Subspace and Factorial Discriminative Graphical Model," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, Minnesota, USA, 2007, pp. 1-8.

[8] R. M. Neal and G. E. Hinton, "A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants," in *Learning in graphical models*, M. I. Jordan, Ed., ed: MIT Press, 1999, pp. 355-368.

[9] M. I. Jordan, Z. Ghahramani, T. Jaakkola, and L. K. Saul, "An Introduction to Variational Methods in Graphical Models," *Machine Learning* vol. 37, pp. 183-233, 1999.

[10] M. Raptis, M. Bustreo, and S. Soatto, "Time Warping Under Dynamic Constraints," in *Workshop on Dynamical Vision in conjunction with ICCV*, Rio de Janeiro, Brazil, 2007.

[11] M. Mackay, R. G. Fenton, and B. Benhabib, "A real-time visual action-recognition framework for time-varying-geometry objects," in *IEEE Conf. on Automation Science and Engineering (CASE)*, 2010, pp. 922-927.

[12] N. Li and D. Xu, "Action Recognition using Weighted Three-State Hidden Markov Model," in *9th Intl. Conf. on Signal Processing (ICSP)*, Beijing, China, 2008, pp. 1428-1431.

[13] M. Ahmad and S.-W. Lee, "HMM-Based Human Action Recognition using Multiview Image Sequences," in *Intl. Conf. on Pattern Recognition (ICPR)*, Hong Kong, 2006, pp. 263-266.

[14] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior Recognition via Sparse Spatio-Temporal Features," in *2nd Joint*

IEEE Intl. Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, Beijing, China, 2005, pp. 65-72.

[15] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing Human Actions: A Local SVM Approach," in *Intl. Conf. on Pattern Recognition (ICPR) Volume 3*, Cambridge, UK, 2004, pp. 32-36.

[16] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld, "Learning Realistic Human Actions from Movies," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska, 2008, pp. 1-8.

[17] R. Vezzani, D. Baltieri, and R. Cucchiara, "HMM based action recognition with projection histogram features," in *20th Intl. Conf. on Pattern Recognition*, Contests Reports, Istanbul, Turkey, 2010, pp. 286-293.

[18] F. Martinez-Contreras, C. Orrite-Urunuela, E. Herrero-Jaraba, H. Ragheb, and S. A. Velastin, "Recognizing Human Actions using Silhouette-Based HMM," in *IEEE Intl. Conf. on Advanced Video and Signal Based Surveillance (AVSS)*, Genoa, Italy, 2009, pp. 43-48.

[19] O. Brdiczka, M. Langet, J. Maisonnasse, and J. L. Crowley, "Detecting Human Behavior Models From Multimodal Observation in a Smart Home," *IEEE Transactions on Automation Science and Engineering (TASE)*, vol. 6, pp. 588-597, 2009.

[20] N. İkizler and D. A. Forsyth, "Searching for Complex Human Activities with No Visual Examples," *Int. Journal of Computer Vision (IJCV)*, vol. 80, pp. 337-357, 2008.

[21] I. Laptev and T. Lindeberg, "Space-Time Interest Points," in *IEEE Intl. Conf. on Computer Vision (ICCV) Volume 1*, 2003, pp. 432-439.

[22] Y. Shen and H. Foroosh, "View-Invariant Action Recognition using Fundamental Ratios," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Anchorage, Alaska, 2008, pp. 1-6.

[23] H. Fujiyoshi and A. J. Lipton, "Real-Time Human Motion Analysis by Image Skeletonization," in *4th IEEE Workshop on Applications of Computer Vision (WACV)*, Princeton, New Jersey, USA, 1998, pp. 15-21.

[24] H.-S. Chen, H.-T. Chen, Y.-W. Chen, and S.-Y. Lee, "Human Action Recognition using Star Skeleton," in *4th ACM Intl. Workshop on Video Surveillance and Sensor Networks*, Santa Barbara, California, USA, 2006, pp. 171-178.

[25] M. A. Ferrer, I. G. Alonso, and C. M. Travieso, "Influence of Initialisation and Stop Criteria on HMM Based Recognisers," *Electronics Letters*, vol. 36, pp. 1165-1166, Jun. 2000.

[26] N. Liu, R. I. A. Davis, B. C. Lovell, and P. J. Kootsookos, "Effect of Initial HMM Choices in Multiple Sequence Training for Gesture Recognition," in *Intl. Conf. on Information Technology: Coding and Computing - Volume 2*, 2004, p. 608.

[27] D. T. Toledano, J. G. Villardebo, and L. A. H. Gomez, "Initialization, Training, and Context-Dependency in HMM-Based Formant Tracking," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, pp. 511-523, Mar. 2006.

[28] I. Ayllón Clemente, M. Heckmann, and B. Wrede, "Incremental word learning: Efficient HMM initialization and large margin discriminative adaptation," *Speech Communication*, vol. 54, pp. 1029-1048, 2012.

[29] K. Murphy. (1998 (last updated on 8 Jun. 2005)). *Hidden Markov Model Toolbox for Matlab*. Available at: http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

[30] J. A. Hartigan, *Clustering algorithms*. New York: John Wiley & Sons, Inc., 1975.

[31] Kingston. University. (2008 (last updated on 12 Apr. 2011)). *The MuHAVi action video dataset.* Available at: http://dipersec.king.ac.uk/MuHAVi-MAS/

[32] M. Basseville and I. V. Nikiforov, *Detection of abrupt changes: theory and applications*: Prentice Hall, 1993.

[33] H. P. Kriegel, P. Kroger, M. Renz, and S. Wurst, "A generic framework for efficient subspace clustering of high-dimensional data," in *5th IEEE International Conference on Data Mining*, 2005, pp. 250-257.

[34] H. Liu and R. Setiono, "Incremental Feature Selection," *Applied Intelligence*, vol. 9, pp. 217-230, 1998.

[35] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as Space-Time Shapes," in *IEEE Intl. Conf. on Computer Vision (ICCV) Volume 2*, Beijing, China, 2005, pp. 1395-1402.

[36] S. Huwer and H. Niemann, "2D-Object Tracking Based on Projection-Histograms," in *European Conf. on Computer Vision (ECCV) Volume I*, 1998, pp. 861-876.

[37] Z. Moghaddam and M. Piccardi, "Human Action Recognition with MPEG-7 Descriptors and Architectures," in *1st ACM Intl. Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Streams (ARTEMIS)*, Firenze, Italy, 2010, pp. 63-68.

[38] M. H. Nguyen, Z.-Z. Lan, and F. D. l. Torre, "Joint segmentation and classification of human actions in video," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3265-3272.

[39] S. Kullback and R. A. Leibler, "On Information and Sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 79-86, 1951.

[40] S. J. Roberts and W. D. Penny, "Variational Bayes for generalized autoregressive models," *IEEE Transactions on Signal Processing*, vol. 50, pp. 2245-2257, 2002.

[41] J. Zhang, "Divergence Function, Duality, and Convex Analysis," *Neural Computing*, vol. 16, pp. 159-195, Jan. 2004 2004.

[42] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 1-27, 2011.