Real-time on-line learning of transformed hidden Markov models from video

Nemanja Petrovic
Beckman Institute
University of Illinois

Nebojsa Jojic
Microsoft Research

Brendan J. Frey
University of Toronto

Thomas S. Huang
Beckman Institute
University of Illinois

Abstract

The transformed hidden Markov model is a temporal model that captures three typical causes of variability in video - scene/object class, appearance variability within the class, and image motion. In our previous work, we showed that an exact EM algorithm can jointly learn the appearances of multiple objects and/or poses of an object, and track the objects or camera motion in video, starting simply from random initialization. As such, this model can serve as a basis for both video clustering and object tracking applications. However, the original algorithm requires a significant amount of computation that renders it impractical for video clustering and its off-line nature makes it unsuitable for real-time tracking applications. In this paper, we propose a new, significantly faster, on-line learning algorithm that enables real-time clustering and tracking. We demonstrate that the algorithm can extract objects using the constraints on their motion and also perform tracking while the appearance models are learned. We also demonstrate the clustering results on an example of typical unrestricted personal media - the vacation video.

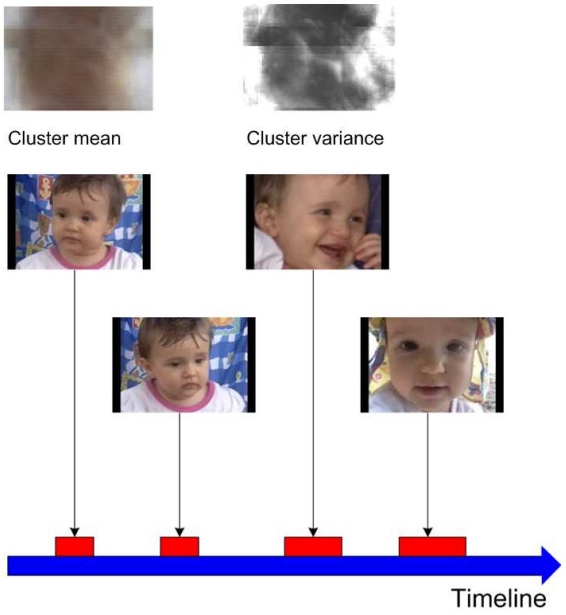

Figure 1: Application of THMM for clustering a 20-minute long vacation video (see also Fig 7 and 6). The mean (best seen in color) and the variance of one of the 43 clusters in the Gaussian mixture are shown in the upper row. The central part of the mean contains and outline of a baby's face in skin color. For this region, the variance is low, while the various backgrounds against which the face appeared are modeled with high variance around the face. Lengths and positions of four video segments found in the first 9 minutes of the video are illustrated on the timeline. These 9 minutes were captured over the period of several days.

1 Introduction

In our previous work, we introduced transformed mixtures of Gaussians (TMG) [1, 2], and their temporal extensions for video analysis, transformed hidden Markov models (THMM) [3]. These algorithms perform joint normalization and clustering of the data. Transformation-invariant clustering models are suitable for video clustering, because they account for the variability in appearance and transformation in the objects and scenes. Therefore, one application of TMG/THMM is video clustering and indexing (Fig. 1).

A frequently mentioned drawback of the transformation-invariant clustering methods is the computational burden of searching over all transformations. In order to normalize for translations of an object over the cluttered background in video sequences, a large number of possible translational shifts should be considered. For exam-

ple, there are $M \times N$ possible integer shifts in an $M \times N$ pixel image. Since the computation time is proportional to the number of pixels and the number of transformations, $\mathcal{O}(M^2N^2)$ operations are used for inference, for each component in the Gaussian mixture. It takes one hour per iteration of the batch EM algorithm to cluster a 40-second long 44x28 pixel sequence into 5 clusters [2].

The temporal extension of the TMG - transformed hidden Markov models (THMM) - use a hidden Markov chain to capture temporal coherence of the video frames. The size of the state space of such an HMM is CMN where C is the number of components in the Gaussian mixture, and MN is the number of translations considered. In [3], the forward-backward algorithm is used to estimate the transition probabilities and the parameters of a THMM, but it adds additional computational time to the TMG, because the transition matrix of the transformations is large. It is also numerically unstable, due to the large number of state-space sequences $(CMN)^T$ for a C-class model of $M \times N$ frames), and the high dimensionality of the data. Only a few state-space paths carry a significant probability mass, and the observation likelihood has a very high dynamic range due to the number of pixels modeled in each sample. This makes the forward-backward algorithm sensitive to the machine precision issues, even when the computation is done in the log domain.

To tackle the computational burden of shift-invariant models, in the past work [4], we proposed to reduce all computationally expensive operations to image correlations in the E step and convolutions with the probability maps in the M step, which made the computation efficient in the Fourier domain. There, the complexity of repeatedly evaluating the likelihood at each stage through I iterations of EM is of the order of $O(CIMN \log(MN))$, thousands of times faster than the technique in [2]. The issues present in the temporal model, THMM, however, still remained.

In this paper, we explore the structure of the model and the structure of the video to derive learning algorithms that run at real-time, which is up to ten thousand times faster than the method in [3].

2 Learning THMM using a Variational Approximation and the M-paths Viterbi algorithm

Under the THMM model, frames in the video sequence are generated from a probability model Fig. 2. The probability density of the vector of pixel values $\mathbf{z}$ for the latent image corresponding to the cluster c is

$$p(\mathbf{z}|c) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_c, \boldsymbol{\Phi}_c), \quad (1)$$

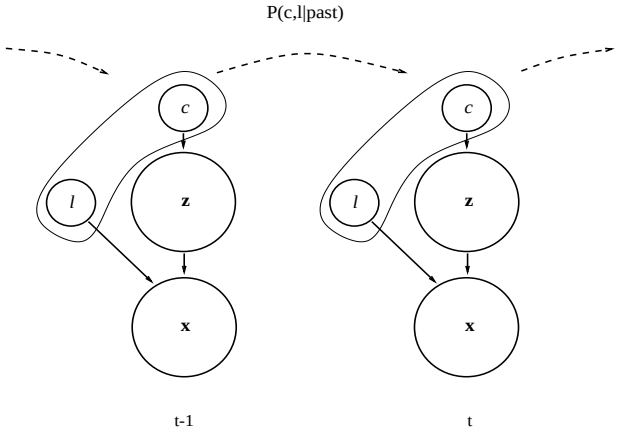

Figure 2: Transformed Hidden Markov Model. Pair $c - \mathbf{z}$ is Gaussian mixture. Gaussian mixture class index (c), and translation index (ℓ) are together the state of an HMM. Observation $\mathbf{x}$ is obtained by translating latent image $\mathbf{z}$ by translation indexed by ℓ.

where $\boldsymbol{\mu}_c$ is the mean of the latent image $\mathbf{z}$, and $\boldsymbol{\Phi}_c$ is a diagonal covariance matrix that specifies the variability of each pixel in the latent image. The probability density of the vector of pixel values $\mathbf{x}$ for the image corresponding to transformation ℓ and latent image $\mathbf{z}$ is

$$p(\mathbf{x}|\ell, \mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\Gamma}_\ell \mathbf{z}, \boldsymbol{\Psi}), \quad (2)$$

where $\boldsymbol{\Psi}$ is a diagonal covariance matrix that specifies the noise on the observed pixels.

The joint likelihood of a single video frame $\mathbf{x}$ and latent image $\mathbf{z}$, given the state of the Markov chain $s \equiv (c, \ell)$, is

$$p(\mathbf{x}, \mathbf{z}|s = (c, \ell)) = \mathcal{N}(\mathbf{x}; \boldsymbol{\Gamma}_\ell \mathbf{z}, \boldsymbol{\Psi}) \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_c, \boldsymbol{\Phi}_c) \quad (3)$$

Note that the distribution over $\mathbf{z}$ can be integrated out in the closed form

$$p(\mathbf{x}|s = (c, \ell)) = \mathcal{N}(\mathbf{x}; \boldsymbol{\Gamma}_\ell \boldsymbol{\mu}_c, \boldsymbol{\Gamma}_\ell \boldsymbol{\Phi}_c \boldsymbol{\Gamma}'_\ell + \boldsymbol{\Psi}), \quad (4)$$

and then from (3) and (4)

$$p(\mathbf{z}|\mathbf{x}, c, \ell) = \mathcal{N}(\mathbf{z}; \boldsymbol{\Omega}_{c\ell} \boldsymbol{\Gamma}'_\ell \boldsymbol{\Psi}^{-1} \mathrm{x}_t + \boldsymbol{\Omega}_{c\ell} \boldsymbol{\Phi}_c^{-1} \boldsymbol{\mu}_c, \boldsymbol{\Omega}_{c\ell}) \quad (5)$$

where $\boldsymbol{\Omega}_{c\ell} = (\boldsymbol{\Phi}_c^{-1} + \boldsymbol{\Gamma}'_\ell \boldsymbol{\Psi} \boldsymbol{\Gamma}_\ell)^{-1}$.

The joint log likelihood of a video sequence $\mathbf{X} = \{\mathbf{x}_t\}_{t=1,\ldots,T}$, hidden states of THMM S, and latent images $\mathbf{Z}$ is

$$\log p(\mathbf{X}, \mathbf{Z}, S) = \log \pi_{c\ell} + \sum_{t=1}^{T} \log p(\mathbf{x}_t, \mathbf{z}_t | s_t) + \sum_{t=1}^{T-1} \log a_{s_t, s_{t+1}} \quad (6)$$

The initial state probabilities are depicted as $\pi_{c\ell}$. The statistical properties of the sequence dynamics are captured in parameters $a_{s_t,s_{t+1}} = p(s_{t+1}|s_t)$. It is reasonable to assume that class index c at time $t+1$ depends only on class index at time t, whereas position index ℓ depends both on previous position and class indices due to the different motion patterns of different objects in the scene. Hence $p(\ell_{t+1}, c_{t+1}|\ell_t, c_t) = p(\ell_{t+1}|\ell_t, c_t)p(c_{t+1}|c_t)$. Furthermore, the transformation transition coefficients can be heavily constrained by choosing the small motion, or the motion in certain direction for given class.

Instead of maximizing log-likelihood of the data, we introduce auxiliary probability density function over hidden variables, $q(\ell, c, \mathbf{z})$, and using Jensen's inequality obtain the lower bound on log-likelihood [6]

$$\log p(\mathbf{X}) = \log \sum_{\{c,\ell\}} \int_{\mathbf{z}} d\mathbf{z}\; p(\mathbf{X}, \{\mathbf{z}, \mathbf{c}, \ell\}) \geq$$
$$\sum_{\{c,\ell\}} \int_{\mathbf{z}} d\mathbf{z}\; q(\{\mathbf{z}, c, \ell\}) \log p(\mathbf{X}, \{\mathbf{z}, c, \ell\}) -$$
$$\sum_{\{c,\ell\}} \int_{\mathbf{z}} d\mathbf{z}\; q(\{\mathbf{z}, c, \ell\}) \log q(\{\mathbf{z}, c, \ell\}) \qquad (7)$$

The second term in the lower bound is the entropy of q and it does not depend on the model parameters. The notation $\{c, \ell\}$ depicts the series of all transformation indices $\{c_t, \ell_t\}, t = 1, \ldots, T$. We implicitly understand that the variational posterior q depends on $\mathbf{X}$. Distribution q can be factored as $q(\{c, \ell\})q(\{\mathbf{z}\}|\{c, \ell\})$, where the first term corresponds to the distribution over states of an HMM.

Therefore, the lower bound on log likelihood is

$$F = \sum_{\{c,\ell\}} \int_{\{\mathbf{z}\}} d\mathbf{z}\; q(\{c, \ell\})q(\{\mathbf{z}\}|\{c, \ell\}) \times$$
$$\big[\log \pi_{c\ell} + \sum_{t=1}^{T} \log p(\mathbf{x}_t, \mathbf{z}_t|c_t, \ell_t) +$$
$$\sum_{t=1}^{T-1} \log p(c_{t+1}, c_t) \log(\ell_{t+1}|\ell_t, c_t)\big] \qquad (8)$$

Learning can now be defined as optimizing the above bound with respect to the parameters of the posterior q and the model parameters.

2.1 Inference (posterior optimization)

By integrating out the hidden variable **z**, the model reduces to a standard HMM with state $s = (c, \ell)$ and the Gaussian observation likelihood given by Eq. (4). Then the posterior $q(\{c, \ell\})$ can be computed exactly using the forward-backward algorithm as in [3]. However, as mentioned in the Introduction, the cost of this operation is not justified as most of the $(CMN)^T$ state paths in the trellis actually have very low probability. Furthermore, the range of the observation likelihood and the size of the state space cause significant problems with numerical precision for sequences longer than a few seconds, even when the path strengths are stored in the log domain. In this paper, we approximate the posterior $q(\{c, \ell\})$ over all possible paths as the mixture of a small number (M) of possible state sequences

$$q(\{c, \ell\}) = \sum_{m=1}^{M} r_m \delta(\{c, \ell\} - \{\hat{c}, \hat{\ell}\}^{(m)}), \qquad (9)$$

where $\sum_{m=1}^{M} r_m = 1$.

One can easily show that to optimize the bound F with respect to the M paths and their strength, it is necessary to find the M best paths in the trellis and set r_m parameters proportional to their likelihoods.

For a given state-space sequence, the distribution $q(\mathbf{z}_t|c_t, \ell_t)$ over the latent image at each time step t can be performed exactly as in Eq. (5) for all M paths.

2.2 Parameter optimization

Defining $u_{c_1c_2} = p(c_{t+1} = c_2|c_t = c_1)$, and $v^{c_1}_{\ell_1\ell_2} = p(\ell_{t+1} = \ell_2|\ell_t = \ell_1, c_t = c_1)$, and finding the constrained derivatives of the bound F

$$0 = \frac{\partial}{\partial u_{c_1c_2}} \Big[\sum_{\{c\},\{\ell\}} \int_{\{\mathbf{z}\}} d\mathbf{z}\; q(\{\mathbf{z}\}|\{c, \ell\}) \times$$
$$q(\{c, \ell\}) \sum_{t=1}^{T-1} u_{c_tc_{t+1}} - \lambda\big(1 - \sum_{c_{t+1}=1}^{C} u_{c_tc_{t+1}}\big)\Big] \qquad (10)$$

gives the optimal transition coefficients assuming the distribution $q(\{x, \ell\}))$

$$\tilde{u}_{c_1c_2} = \frac{\sum_{t=1}^{T-1} q(c_t = c_1, c_{t+1} = c_2)}{\sum_{t=1}^{T-1} q(c_t = c_1)}$$
$$= \frac{\sum_{t=1}^{T-1} \sum_{m=1}^{M} r_m \delta(\hat{c}_t^{(m)} - c_1)\delta(\hat{c}_{t+1}^{(m)} - c_2)}{\sum_{t=1}^{T-1} \sum_{m=1}^{M} r_m \delta(\hat{c}_t^{(m)} - c_1)} \qquad (11)$$

and

$$\tilde{v}^{c_1}_{\ell_1\ell_2} = \frac{\sum_{t=1}^{T-1} q(\ell_t = \ell_1, \ell_{t+1} = \ell_2, c_t = c_1)}{\sum_{t=1}^{T-1} q(\ell_t = \ell_1, c_t = c_1)}$$
$$= \frac{\sum_{t=1}^{T-1} \sum_{m=1}^{M} r_m \delta(\hat{\ell}_t^{(m)} - \ell_1)\delta(\hat{\ell}_{t+1}^{(m)} - \ell_2)\delta(\hat{c}_t^{(m)} - c_1)}{\sum_{t=1}^{T-1} \sum_{m=1}^{M} r_m \delta(\hat{\ell}_t^{(m)} - \ell_1)\delta(\hat{c}_t^{(m)} - c_1)} \qquad (12)$$

Intuitively, we expect the matrix of coefficients $\tilde{u}_{c_1c_2}$ to have large diagonal elements, since we favor the new

frame to remain in the same cluster as the previous. Also, it is possible to severely constrain transformation transition coefficients $\tilde{v}_{\ell_1\ell_2}^{c_1}$ and in that fashion set the motion prior for the class, for example, favoring the small motion or the motion in one direction.

Finding the derivatives with respect to the cluster means, we get

$$\sum_{t=1}^{T}\sum_{\{\ell,c\}:c_t=k} q(\{c,\ell\})\int_{\mathbf{z}_t} d\mathbf{z}_t\; q(\{\mathbf{z}_t\}|\{c,\ell\})\times \left(-\frac{1}{2}\mathbf{\Phi}_k^{-1}(\mathbf{z}_t-\boldsymbol{\mu}_k)\right)=0 \quad (13)$$

Due to the nature of the Markov chain, given the pair (c_t,ℓ_t), $\mathbf{z}_t$ does not depend on any other (c,ℓ), nor on any other $\mathbf{x}$ apart from $\mathbf{x}_t$. Therefore, $\mathrm{E}(\mathbf{z}_t|\mathbf{X},\{c,\ell\}) = \mathrm{E}(\mathbf{z}_t|\mathbf{x}_t,c_t,\ell_t)$, where $\mathrm{E}(\mathbf{z}_t|\mathbf{x}_t,c_t,\ell_t)$ is given by (5), and thus,

$$\mathbf{\Phi}_k^{-1}\sum_{t=1}^{T}\sum_{\ell_t} q(c_t=k,\ell_t)\mathbf{E}[\mathbf{z}|\mathbf{x}_t,c_t=k,\ell_t] = \mathbf{\Phi}_k^{-1}\sum_{t} q(c_t=k)\boldsymbol{\mu}_k. \quad (14)$$

Subsequently, if the posterior is known, the means $\boldsymbol{\mu}_k$ can be set in the batch EM update,

$$\tilde{\boldsymbol{\mu}}_k = \frac{\sum_{t=1}^{T}\sum_{\ell_t} q(c_t=k,\ell_t)\mathrm{E}[\mathbf{z}_t|\mathbf{x}_t,c_t=k,\ell_t]}{\sum_{t=1}^{T} q(c_t=k)} \quad (15)$$

where the summation in $\sum_{\ell_t}$ is only over the paths that pass through class k at time t. Even though the number of transformations is equal to the number of pixels we limit the search only to the transformations that yield a non-zero term $q(c_t=k,\ell_t)$, which in turn is computed by simply looking-up those of M paths in q that pass through class k at time t

$$q(c_t=k,\ell_t)=\sum_{m=1}^{M} r_m\delta(\hat{c}_t^{(m)}-k)\delta(\hat{\ell}_t^{(m)}-\ell_t), \quad (16)$$

and

$$q(c_t=k)=\sum_{m=1}^{M} r_m\delta(\hat{c}_t^{(m)}-k) \quad (17)$$

Similarly, covariance matrix update is

$$\tilde{\mathbf{\Phi}}_k = \frac{1}{\sum_{t=1}^{T} q(c_t=k)}\sum_{t=1}^{T}\sum_{\ell_t} q(c_t=k,\ell_t)\times [(\mathrm{E}[\mathbf{z}_t|\mathbf{x}_t,c_t=k,\ell_t]-\boldsymbol{\mu}_k)\circ(\mathrm{E}[\mathbf{z}_t|\mathbf{x}_t,c_t=k,\ell_t]-\boldsymbol{\mu}_k)+\mathbf{\Omega}_{c\ell}] \quad (18)$$

The variational approximation of the posterior distribution over the states of an HMM with M best sequences significantly reduces the computational burden in EM update.

3 Recursive, on-line EM

While improving somewhat the computational efficiency and solving the problems that the exact learning algorithm of [3] had with the machine precision, the M-best paths learning described in the previous section still suffers from two drawbacks: a) the need to preset the number of classes C, and b) the need to iterate. For a typical 20-minute sequence, the needed number of classes C can range anywhere from five to fifty (see Fig. 6 and Fig. 7), and more extreme values are possible, as well. While the number of iterations needed to converge also depends slightly on the number of classes, it is typically sufficient to run the algorithm for ten or twenty iterations. The computational cost is proportional both to the number of iterations and the number of classes, but the structure of realistic video allows development of more efficient algorithms. Frames in video typically come in bursts of a single class which in principle means that the algorithm does not need to test all classes against all frames all the time, and also that there is an abundance of data to learn the class model from, thus opening room for an on-line learning algorithm that adapts to the new data and slowly forgets the old.

In this section we propose an on-line algorithm that passes through the data only once and which introduces new classes as the new data is observed. Most of the time, the algorithm is simply inferring the transformations under only one class, and only during the possible class transition periods it evaluates all classes. This makes the algorithm's performance equivalent to a single-iteration, single-class THMM learning, which for the typical numbers we gave above for a twenty-minute sequence leads to an improvement of a factor 300 against ten iterations of the batch learning of a 30-class THMM.

To derive the on-line learning algorithm, we reconsider Eq. (14) and define

$$S_{k,T} \triangleq \mathbf{\Phi}_k^{-1}\sum_{t=1}^{T}\sum_{\ell_t} q(c_t=k,\ell_t)\mathbf{E}[\mathbf{z}|\mathbf{x}_t,c_t=k,\ell_t] \quad (19)$$

and

$$R_{k,T} \triangleq \mathbf{\Phi}_k^{-1}\sum_{t=1}^{T} q(c_t=k). \quad (20)$$

Then, batch update of $\boldsymbol{\mu}_k$ using T data samples is $\boldsymbol{\mu}_k^{(T)} = R_{k,T}^{-1}S_{k,T}$, as in (15). If we rewrite (14) for $T+1$ data samples

$$S_{k,T}+\mathbf{\Phi}_k^{-1}\sum_{\ell_{T+1}} q(c_{T+1},\ell_{T+1})\mathbf{E}[\mathbf{z}|\mathbf{x}_{T+1},c_{T+1},\ell_{T+1}] = R_{k,T}\boldsymbol{\mu}_k + \mathbf{\Phi}_k^{-1}q(c_{T+1}=k)\boldsymbol{\mu}_k \quad (21)$$

Figure 3: Three frames from the Walk sequence, corresponding to the beginning, middle part, and the end of the sequence.

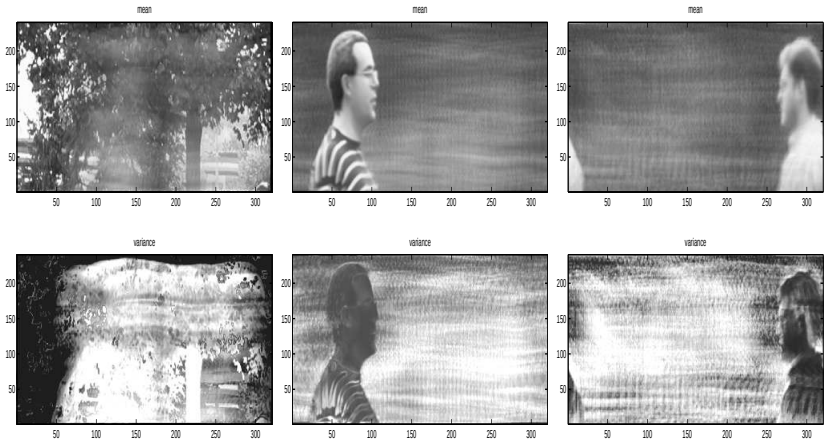

Figure 4: Means and variances for three classes learned using variational THMM for the Walk sequence. For variances, black implies very low variance and white very high variance. First column corresponds to learned background, second to the object moving to the right, and third to the object moving to the left.

Multiplying (21) from the left with R_T^{-1}

$$\boldsymbol{\mu}_k^{(T)} + R_{k,T}^{-1}\boldsymbol{\Phi}_k^{-1}\sum_{\ell_{T+1}} q(c_{T+1},\ell_{T+1})\mathbf{E}[\mathbf{z}|\mathbf{x}_{T+1},c_{T+1},\ell_{T+1}] = \left[\mathrm{I} + R_{k,T}^{-1}\boldsymbol{\Phi}_k^{-1}q(c_{T+1}=k)\right]\boldsymbol{\mu}_k^{(T+1)}, \quad (22)$$

and assuming the term in the square brackets on the RHS is close to 1, and using the matrix equivalents of $\frac{1}{1+x} \approx 1-x$ and $\frac{x}{1+x} \approx x$

$$\boldsymbol{\mu}_k^{(T+1)} = [\mathrm{I} - R_{k,T}^{-1}\boldsymbol{\Phi}_k^{-1}q(c_{T+1}=k)]\boldsymbol{\mu}_k^{(T)} + R_{k,T}^{-1}\boldsymbol{\Phi}_k^{-1}\sum_{\ell_{T+1}} q(c_{T+1},\ell_{T+1})\mathbf{E}[\mathbf{z}|\mathbf{x}_{T+1},c_{T+1},\ell_{T+1}] \quad (23)$$

The statistics $R_{k,T}$ as it is defined is increasing and ultimately diverges to infinity. It could be rewritten in terms of its time average. If we define the short-time average as

$$\tilde{R}_{k,T,\Delta T} \triangleq \frac{1}{\Delta T}\sum_{t=T-\Delta T+1}^{T} \boldsymbol{\Phi}_k^{-1}q(c_{T+1}=k), \quad (24)$$

then

$$R_{k,T} = \frac{T}{T}\sum_{t=1}^{T}\boldsymbol{\Phi}_k^{-1}q(c_{T+1}=k) = T\tilde{R}_{k,T,T} = \frac{1}{\alpha}\tilde{R}_{k,T,T} \quad (25)$$

where we define $\alpha = \frac{1}{T}$ to be the learning rate. By taking $\alpha = \frac{1}{\Delta T}$ instead, we perform learning with forgetting.

It is not difficult to prove using the definition that $\tilde{R}_{k,T,\Delta T}$ is updated using the recursion

$$\tilde{R}_{k,T,\Delta T} = (1-\alpha)\tilde{R}_{k,T-1,\Delta T} + \alpha\boldsymbol{\Phi}_k^{-1}q(c_T=k) \quad (26)$$

Note that this update is different than the update based on taking a step in the direction of the bound or likelihood gradient, as suggested in [7]. The gradient-based approach produces small steps for unlikely classes, thus rendering learning of multi-class models slow. This does not happen in the batch learning, due to the normalization with the *total* responsibility of the class k in the video. Thus, even if the class was initialized far from the data, after the very first iteration of EM it will jump close to it, as the update becomes an average of the data points. The gradient-based on-line learning, on the other hand, due to the small value of the posterior for each data point, moves the parameters very slowly.

In our update, the additional quantity $\tilde{R}_{k,T,\Delta T}$ plays the role of the average responsibility of the class k in the previously seen frames. This quantity is also tracked through time in addition to the model parameter, and so rather than making a step scaled by the responsibility of

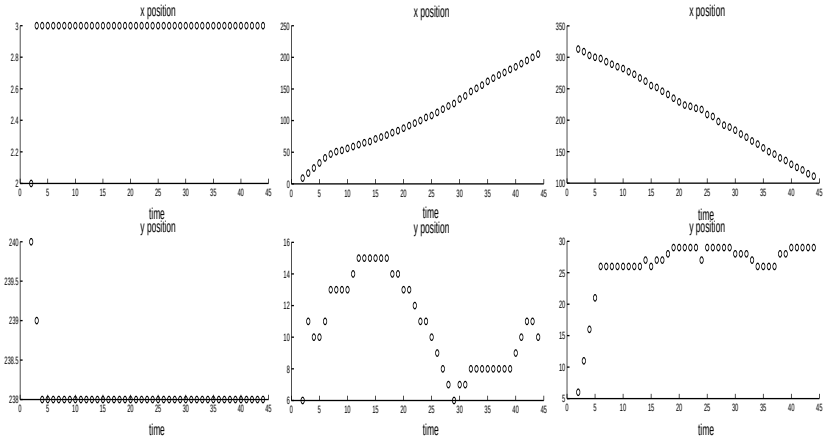

Figure 5: The most likely object position for the three clusters in the 44-frame long 240x320 Walk sequence. Translations are defined as being wrapped around the image boundaries, therefore the downward shift of 239 corresponds to the upward shift of 1. The upper row represents tracking of the horizontal coordinate, while the lower row represents the tracking of the vertical coordinate. Class 2 and 3 slightly oscillate in the vertical direction, hence the detected displacement of a few pixels.

the class, what matters in the update equation is the ratio of the class responsibility for the new data point and the average class responsibility for the previously seen points.

Substituting (25) in (23) we get the final update rule for $\boldsymbol{\mu}_k$ (and similarly for $\boldsymbol{\Phi}_k$)

$$\boldsymbol{\mu}_k^{(T+1)} = [\mathrm{I} - \alpha \tilde{R}_{k,T,\Delta T}^{-1} \boldsymbol{\Phi}_k^{-1} q(c_{T+1}{=}k)] \boldsymbol{\mu}_k^{(T)} + \alpha \tilde{R}_{k,T,\Delta T}^{-1} \boldsymbol{\Phi}_k^{-1} \sum_{\ell_{T+1}} q(c_{T+1}, \ell_{T+1}) \mathbf{E}[\mathbf{z} | \mathbf{x}_{T+1}, c_{T+1}, \ell_{T+1}] \quad (27)$$

where the change of $\tilde{R}_{k,T,\Delta T}$ is governed by (26). Note that setting $\alpha = \frac{1}{T}$ we achieve the same update as in Eq.(15), when the posterior is fixed and exact. In on-line learning we set ΔT to be smaller, and also allow the posterior to change. In our experiments we kept $\alpha = 0.01$, thus updating parameters using only approximately previous 100 data samples (frames).

Note that in Eq. (26) the average class responsibility is combined with the current class variability model (by keeping $\boldsymbol{\Phi}_k^{-1}$). In batch learning, since the covariance matrix is assumed fixed in one iteration, it can be divided out of the update equation (14) for the mean. However, since we change the parameters through the time, including $\boldsymbol{\Phi}_k^{-1}$ in (26) helps refine parts of the class mean at different speeds depending on the uncertainty.

To deal with the introduction of the new classes in the video, we observe first of all that the problem is ill-posed, i.e., it needs to be constrained. For example, model selection in on-line learning is sometimes constrained by specifying the prior on the model parameters [10]. Since our goal is data summarization, we constrain the class number selection by specifying the lowest allowable data likelihood. Such constraint simply states that all data needs to be explained reasonably well (with the likelihood above some threshold), while the classes should be introduced as rarely as possible. The temporal structure in the THMM and the structure of a realistic video (alternating bursts of frames from each class) suggest that if the likelihood of the current frame under the class associated with the previous frame is reasonably high, there is no need to evaluate the likelihood under the previous classes, as the current class provides a good model. So, we use two thresholds on the likelihood $\gamma_1 > \gamma_2$. When the log-likelihood of the observed frame under the class from the previous frame is above γ_1, we classify it as belonging to the same class. When this likelihood is smaller than γ_1, the full posterior is computed and the likelihood of the data re-estimated, leading often to classifying the frame as belonging to some other, previously seen class. However, if the data likelihood under the full model is still lower than γ_2, a new class is introduced and initialized to the problematic frame. This is similar to stability-plasticity dillema of Grossber's ART [8], whose unsupervised version iteratively clusters data, and introduces a new cluster if none of the existing clusters can explain the new sample well.

This approach guarantees that the likelihood of the data will be limited from below, since during the learning the likelihood of the current frame never drops below some threshold, and the subsequent possible drop in the likelihood due to the model update is limited by the slow learning rate.

However, due to the sensitivity of the number of introduced classes to the threshold γ_2, it is possible for the single-pass learning algorithm to introduce several

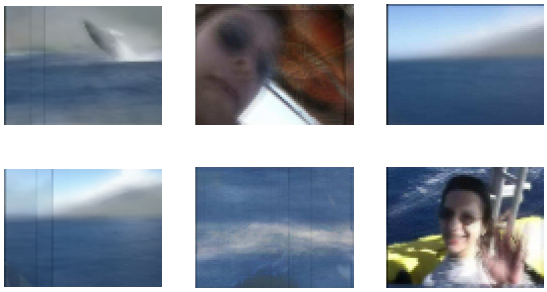

Figure 6: The summary of a 20-minute long whale watching sequence. Interesting events (whale, people) are buried in the video of the ocean and the mountains. The video is clustered into six classes, whereas most of the frames are clustered into clusters 3, rightmost in the top row, and 4, leftmost in the bottom row.

classes modeling similar frames. This can be detected without looking back at the data, by finding the expected likelihood of the frames from one class under the probability model of the other,

$$
\begin{aligned}
&L_{1,2} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} \log p(y_n|c_1) = \\
&\mathrm{E}_{\boldsymbol{\mu}_2,\boldsymbol{\Phi}_2}\big[\log p(y_n|\boldsymbol{\mu}_1,\boldsymbol{\Phi}_1)\big] = \\
&\mathrm{E}_{\boldsymbol{\mu}_2,\boldsymbol{\Phi}_2}\big[-\frac{1}{2}\log|\frac{\boldsymbol{\Phi}_1}{2\pi}| - \frac{1}{2}(y_n-\boldsymbol{\mu}_1)'\boldsymbol{\Phi}_1^{-1}(y_n-\boldsymbol{\mu}_1)\big] = \\
&\quad -\frac{1}{2}\log|\frac{\boldsymbol{\Phi}_1}{2\pi}| - \frac{1}{2}\mathrm{tr}(\boldsymbol{\Phi}_1^{-1}\boldsymbol{\Phi}_2) - \\
&\qquad \frac{1}{2}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1)'\boldsymbol{\Phi}_1^{-1}(\boldsymbol{\mu}_2-\boldsymbol{\mu}_1) \qquad (28)
\end{aligned}
$$

To achieve a more compact representation, class 1 and 2 in this example can be merged into one when both $L_{1,2}$ and $L_{2,1}$ are larger than the likelihood level required in the video summary. The Gaussians for two classes are merged into one in a straight-forward manner.

4 Experimental Results

4.1 Extracting objects using motion priors

We demonstrate object extraction from the scene by setting motion priors in coefficients $a_{s_t,s_{t+1}} = u_{c_1c_2} \cdot v^{c_1}_{\ell_1\ell_2}$. The difference between two consecutive transformation indices $\ell_t = \ell_1$ and $\ell_{t+1} = \ell_2$ corresponds to the inter-frame motion. The direction or the intensity of the motion for class c_1 can be constrained by setting appropriate elements in $v^{c_1}_{\ell_1\ell_2}$ to zero. We demonstrate the use of motion templates for extracting the background and two objects that move in the different directions. We trained a THMM on a 44 frames long 320×240 Walk sequence (Fig. 3), using M-paths batch learning described in section Section 2. Training was performed at the rate of 3 fps, and it took 8 iterations of EM algorithm to obtain results in Fig. 4. We trained the model using a three-component mixture of Gaussians, and we set the motion priors for each of the components. One of the components was allowed horizontal and vertical shifts of up to one pixel. Second and third components were allowed horizontal shifts of no less than two and no more than eight pixels to the left and right, respectively. The algorithm was able to pull out the two objects and the background scene into three classes (Fig. 4). Without motion priors, both TMG and THMM in its approximate or exact forms, only learn the classes similar to the first class in Fig. 4(the background image), and are never able to track and learn the two people. The motion priors, however, help set the position inference on the right track (Fig. 5), and lead to reasonable template learning results, even though the class means were initialized to random images. The effect of the temporary occlusion of the person in white shirt is reduced in the mean image (Fig. 4), due to the integration with other frames where the person was fully or partially visible. In both person appearance models, the background is suppressed and the class variance is mostly lower on the pixels belonging to the appropriate person. Despite the slight blemishes on the appearance models, for all three classes the tracking is very precise. By modeling the data as a mixture, rather than a composition of the objects, THMM is unable to learn perfect object appearance or perform good segmentation as in [11]. But, THMM can be used as an initial approximation for the layered model, and it is an order of magnitude faster. It can be also used for object tracking, and as a motion detector.

4.2 On-line, real time video clustering

In the first example we use a 20-minute long 90x60, 15 fps color Hawaii sequence. This typical vacation sequence was shot during the period of several days, at different locations, but with the same group of people. We train THMM using on-line learning, starting with the Gaussian mixture with one component. Learning and subsequent class introduction is performed at the average rate of 10 frames per second on 2.2GHz P4. We show learned cluster means in Fig. 7. The commercial video shot detection software, like Microsoft's MovieMaker, usually detects only a few shots in this type of video, as most of the visual diversity in the video is caused by camera wipe rather than by camera cut. Without any clear shot cuts, the traditional software simply represents the video with thumbnails for several long video segments. These thumbnails, however, only show one frame from the shot, hiding all the other content. Our clustering provides a much richer presentation, and beyond the video segmentation task, it groups similar scenes and objects together as illustrated in Fig. 1. After

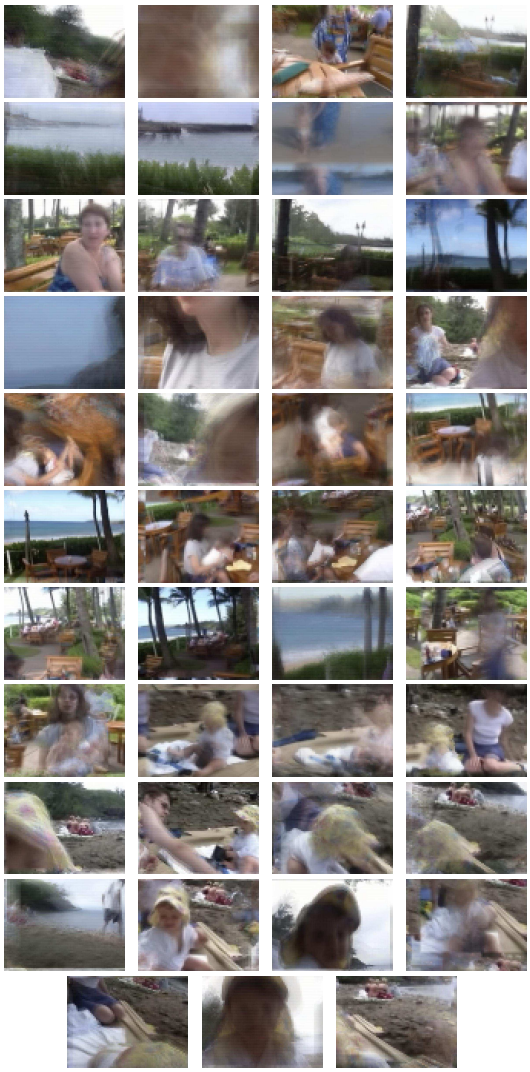

Figure 7: The summary of a 20-minute long Hawaii sequence: cluster means for 43 classes learned using online algorithm at 10 frames per second.

the cluster merging step, our model consists of 43 clusters (Fig. 7).

In the last example, we illustrate the scalability of our approach to class introduction based on the minimum allowable likelihood. In a 20-minute long video from a whale watching trip, most of the video is jerky and contains the same scene consisting of the mountains and the ocean, while the exciting segments, containing whale breaches and people, are less than a minute long in total. In this sequence, our algorithm was able to jointly stabilize the sequence and find nine clusters that were in the end merged in the six clusters shown in Fig. 6. Most of the sequence was explained with only two clusters, while much shorter and content-rich parts were explained by four clusters. After learning a summary like this, it is easy to find interesting shots in the video.

Review [9] surveys the recent studies in the area of content based video analysis. Although there is no agreed standard to compare different algorithms, our approach unifies several stages of video analysis: video partitioning, cut detection, motion characterization, scene representation, and definition of scene similarity measure.

5 Conclusions

We presented a fast variational on-line learning technique for training a transformed hidden Markov model. We demonstrated that learning a realistic video can be performed in real time, a 10000-fold improvement in computation over the method in [3]. We believe these techniques would prove useful in video clustering, summarization and indexing.

References

[1] B. J. Frey & N. Jojic. Transformation-Invariant Clustering Using the EM Algorithm. IEEE Trans.PAMI,25:1,Jan 2003

[2] B. Frey & N. Jojic. Learning mixture models of images and inferring spatial transforms using the EM algorithm. *CVPR 99*, Ft. Collins, Colorado

[3] N. Jojic, N. Petrovic, B. Frey & T. S. Huang. Transformed hidden Markov models: Estimating mixture models and inferring spatial transformations in video sequences. *CVPR 2000*, Hilton Head Island, SC

[4] Frey, B.J. & Jojic, N. Fast, large-scale transformation-invariant clustering. In *Advances in Neural Information Processing Systems 14*. Cambridge, MA: MIT Press 2002

[5] J. K. Wolf, A. M. Viterbi & G. S. Dixon. Finding the best set of K paths through a trellis with application to multitarget tracking. In *IEEE Trans. on Aerospace & Elect. Sys.* pp.287-296, Vol. 25, No. 2, Mar. 1989

[6] R. M. Neal & G. E. Hinton. A new view of the EM algorithm that justifi es incremental, sparse and other variants. In *Learning in Graphical Models*. Kluwer Academic Publishers, Norwell MA, Ed. M. I. Jordan, pp.355-368, 1998

[7] E. Bauer, D. Coller & Y. Singer. Update rules for parameter estimation in Bayesian networks. In *Proceedings of the Thirteenth UAI*. pp.3-13, Providence, RI, 1997

[8] Carpenter, G.A. & Grossberg, S. Learning, Categorization, Rule Formation, and Prediction by Fuzzy Neural Networks. In Chen, C.H., ed. (1996) *Fuzzy Logic and Neural Network Handbook*, McGraw-Hill, pp.1.3-1.45

[9] C.-W. Ngo, T.-C. Pong & H.-J. Zhang Recent Advances in Content-Based Video Analysis. In *International Journal of Image and Graphics*, Vol. 1, No. 3, pp.445-468, 2001

[10] Z. Ghahramani. Online Variational Bayesian Learning. *Slides from talk presented at NIPS workshop on Online Learning*. 2000.

[11] N. Jojic & B. Frey. Learning flexible sprites in video layers. *CVPR 2001*