Efficient Hidden Semi-Markov Model Inference for Structured Video Sequences

David Tweed, Robert Fisher, José Bins, Thor List
University of Edinburgh,
Institute for Perception, Action & Behaviour, School of Informatics,
James Clerk Maxwell Bldg, Kings Bldgs, Mayfield Road, Edinburgh EH9 3JZ
{dtweed,rbf,jbfilho,tlist}@inf.ed.ac.uk

Abstract

The semantic interpretation of video sequences by computer is often formulated as probabilistically relating lower-level *features* to higher-level *states*, constrained by a transition graph. Using *Hidden Markov Models* inference is efficient but time-in-state data cannot be included, whereas using *Hidden Semi-Markov Models* we can model duration but have inefficient inference. We present a new efficient $O(T)$ algorithm for inference in certain HSMMs and show experimental results on video sequence interpretation in television footage to demonstrate that explicitly modelling time-in-state improves interpretation performance.

keywords: computer vision, Hidden Markov models, video behaviour analysis, activity recognition

1. Introduction

Many computer vision tasks, particularly image understanding tasks, boil down to questions about the temporal structures – composed from low-level 'object features' – that occur over a video sequence rather than the individual features themselves. For example, detecting that an individual is 'window-shopping' might be performed by looking for a characteristic pattern of the individual moving at walking speed and then stopping in front of shop windows; there is no simple, directly observable element in any individual frame that signifies the activity window-shopping is occurring. Producing good interpretations requires combining both low-level image features and temporal structure.

The general temporal structure fitting problem assigns a *state* (from a predefined set) to each frame based upon features extracted from both individual frames and runs of adjacent frames. These states often correspond to high-level notions of activity, e.g., walking, window-shopping, fighting, etc. In contrast, features extracted directly from the images are generally relatively low-level and often statistically based, e.g., inter-frame energy or bounding box motion. Even the deterministic features can generally be related to higher-level states only on a probabilistic level, so we want a modelling technique that maximises accuracy of inference by letting us combine the direct probabilistic relationships between features and states with as much other available knowledge (either *a priori* relationships or learned from training data).

We want to determine whether one out of a set of modelled behaviours occurs in the sequence, and if so output a detailed assignment of labels to individual frames. On-line reporting of important events and dynamic self-adjustment by the system, eg, camera servoing, make it desirable that the analysis algorithm performs inference *incrementally* as new frames are observed rather than batch processing complete sequences.

Hidden Markov Models (HMM) methods are traditionally used for this sort of problem, but they do not represent the time-in-state particularly well for video sequence analysis. Hidden Semi-Markov Models (HSMMs) do allow arbitrary distributions but don't generally have the $O(T)$ algorithms needed for practical continuous video analysis. This paper describes a novel algorithm for efficient inference when the temporal distributions satisfy a 'convex monotonicity' property.

Previous work. Algorithms for recognition of behaviour in video tend to be based upon distilling the contents of each frame into symbolic labels, then fitting some form of generative probabilistic model relating high-level behaviours to the symbolic labels. Judgements about which kinds of structure are important for the applications considered lead to different generative models. At the simplest level, [20] describes a system for understanding snooker using a simple HMM after converting the sequence into events such as ball collisions and pottings.

A natural assumption is that there is a 'hierarchy of processes' occurring at different temporal granularities, so that multiple simple models can be trained to recognise each level of the hierarchy, rather than a complex model for the entire process. [18] develops the *Layered HMM* using this

assumption and use it to recognise office behaviour.

Another situation is modelling behaviours composed of interacting subprocesses, eg, [4] develops a *Coupled HMM* and recognises tai-chi actions from limb-tracking data. Interacting processes are particularly important when dealing with behaviour involving multiple people.

Although HMM-based models are most popular for behaviour understanding, there are at least two other approaches. Firstly, structure can be expressed instead using stochastic context free grammars. These are arguably more complicated models but are better for behaviours with extensive dependencies between the next state and states far in the past, eg, [14] models blackjack whilst [16] deals with car park behaviour. ([10] uses a *Variable Length HMM* to deal with varying lengths of dependency on the past, retaining the efficiency of the HMM for short-term dependencies.) Secondly, there are purely logic-based systems for behaviour understanding, eg, [3, 6, 21]. Logical reasoning is used extensively in other areas of AI, but it is open whether it is sufficiently robust for video sequence analysis given the ambiguity in features extracted from images.

For video analysis the time in a state is often much longer than the sampling frequency, whereas a HMM favours shorter times-in-state (as explained in section 2). The HSMM model we use to counteract this is also used by [13] in their behaviour recogniser and [22] in their activity recognition and abnormality detection technique. The HSMM model is examined in extensive detail in [12]. This survey of applications is especially helpful on learning model parameters. However, the algorithms used previously for HSMMs are $O(TD)$, where D is the longest time-in-state allowed; for parametric distributions this is unlimited, leading to an $O(T^2)$ algorithms, effectively limiting the length of video sequence which can be analysed. [15] is a survey of the various HSMM decoding algorithms. This states the fastest general decoding algorithm is given in [23], which tackles the more general problem of inference in the presence of partially missing observation data.

The algorithms developed here are an application of existing dynamic programming algorithms developed for efficient minimal-cost matching of DNA sequences [11, 7]. These solve a class of optimisation problems by exploiting the observation that, whilst in its general form the dynamic programming solution has high computational complexity, the cost functions used in practice are often 'convex' or 'concave', which can be used to give algorithms of lower complexity. The applicability of this dynamic programming approach to HSMMs is non-obvious because they seek to minimise additive costs whereas the basic formulation of probabilistic models is in terms of maximising products. However, the connection emerges when the HSMM model is converted to 'negated log probabilities'.

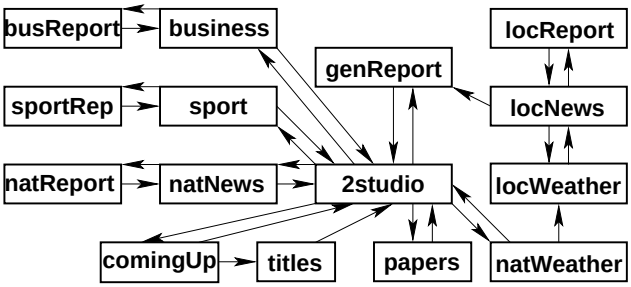


Figure 1: A typical transition graph (here for breakfast TV)

2. Modelling sequence structure using Markov Transition Graphs

In this section we develop the Hidden Semi-Markov Model we will use later, starting with a relatively abstract model and showing how different ways of 'concretising' it lead to Hidden Markov Models and Hidden semi-Markov Models.

We start with a sequence of observed feature vectors $\mathbf{f}_1, \ldots, \mathbf{f}_T$ (eg, whole frames, block-correlation scores, etc) which are extracted from video of a physical scene where a particular behaviour is occurring. We have a set of *states* $\mathcal{S}=\{\mathsf{s}^1, \ldots, \mathsf{s}^S\}$ (using superscripts to enumerate labels) corresponding to the various possible activity states occurring in individual frames. For the problem we deal with, at each timestep t the physical scene is in a particular state s_t, but we can't compute s_t directly from $\mathbf{f}_t$, ie, the ss are *hidden variables*. This is due to observations containing a limited amount of information from the physical scene and, most intractably, imprecise models for scene understanding. However we assume we have *observation models* which compute the probability $P(\mathbf{f}|\mathsf{s})$ that feature vector $\mathbf{f}$ is observed when the system is in state s. The obvious way to get the most likely sequence of states is to pick the state with highest *a posteriori* probability for each frame *independently*. It's clear, however, that if we have more knowledge about the system we can use this to perform more accurate inference.

The most basic structure we can impose on the system is a *transition graph* such as Fig 1 describing allowed transitions between states, in this case derived from TV footage. At its most general, this graph specifies *which* new states are allowed *when* a state change occurs, thus allowing any interpretation where the sequence of states violates the graph to be discarded. A temporal process is called *Markov* when the future state depends only on the *present* state and not on the exact details of the past, for some chosen definition of 'present'; the graph has the Markov property that the new state is constrained *only* by the present state. This is a trade-off: Markov assumptions enable efficient evaluation and, by limiting the number of model parameters, keep the amount of training data needed practical whilst captur-

ing the real world dependencies which are generally most signifi cant. However, some natural constraints effectively cannot be expressed by a transition graph, a typical example being 'moving to state Y can only occur if state X has occurred *at some point* in the past'. Nevertheless, for suitable problems the transition graph can be very effective.

Models based on transition graphs. If, for each state s^i, we know the probability $P(\mathsf{s}^j|\mathsf{s}^i)$ of each transition allowed by the graph, we could incorporate this into the inference process and thus compute a more accurate probability of a given interpretation. To complete this we have to handle remaining in the same state between adjacent timesteps. An obvious solution is to *require* a state transition at each timestep and add 'transition to same state' edges to the behaviour graph with appropriate probabilities; in terms of the Markov assumption this makes *present* mean 'current frame'. The probability of observing features $\mathbf{f}_{1:T}$ with interpretation $\mathsf{s}_{1:T}$ is then

$$P(\mathbf{f}_{1:T}, \mathsf{s}_{1:T}) = P(\mathsf{s}_1)\left[\prod_{t=1}^{T} P(\mathbf{f}_t|\mathsf{s}_t)\right]\left[\prod_{t=2}^{T} P(\mathsf{s}_t|\mathsf{s}_{t-1})\right] \quad (1)$$

$$= P(\mathsf{s}_1)P(\mathbf{f}_1|\mathsf{s}_1)\left[\prod_{t=2}^{T} P(\mathsf{s}_t|\mathsf{s}_{t-1})P(\mathbf{f}_t|\mathsf{s}_t)\right] \quad (2)$$

$$= P(\mathbf{f}_{1:T-1}|\mathsf{s}_{1:T-1})P(\mathsf{s}_T|\mathsf{s}_{T-1})P(\mathbf{f}_T|\mathsf{s}_T) \quad (3)$$

where s_t is the particular state assigned at time t, $a{:}b$ denotes $a, \ldots, b{-}1$ and likewise $x_{a:b}{=}x_a, \ldots, x_{b-1}$. (The prior probability of the initial state $P(\mathsf{s}_1)$ complicates formulae and could be avoided with an artifi cial start state whose transition probabilities encode the prior on the true initial state.) This is the standard *Hidden Markov Model (HMM)* [19]. Eq 1 shows the split into feature- and transition-based terms, whilst Eq 2 & Eq 3 shuffle terms to highlight the Markov property that the state for the new frame depends only on the new observation, the state for the *present timestep* and the overall probability of the state sequence so far.

One answer to the question 'what should one use as the values for the hidden states $\mathbf{s}_{1:T}$ when observation sequence $\mathbf{f}_{1:T}$ is seen?' is the sequence $\mathbf{s}_{1:T}$ which maximises the probability $P(\mathbf{f}_{1:T}, \mathsf{s}_{1:T})$ in Eq 1. This has the advantage over other answers that $\mathbf{s}_{1:T}$ obeys the transition constraints embodied in the transition graph. The recursive structure of Eq 3 is useful because it can be seen that the maximum value of $P(\mathbf{f}_{1:T}, \mathsf{s}_{1:T})$ in Eq 1 *for a particular choice of* s_{T-1} occurs for the choice of $P(\mathbf{f}_{1:T-1}|\mathsf{s}_{1:T-1})P(\mathsf{s}_T|\mathsf{s}_{T-1})$ with maximal value. This means that standard dynamic programming techniques [1, 19] can be used to perform this maiximisation incrementally taking $O(S^2T)$ time overall. This maximisation process is known as *inference* or *decoding*.

However there is one assumption in this model which is not immediately obvious. By direct calculation we can see that the prior probability of staying in state s^i for exactly τ steps is $P(\mathsf{s}^i|\mathsf{s}^i)^\tau(1{-}P(\mathsf{s}^i|\mathsf{s}^i))$. Thus in a HMM the self-transition probability is really a parameter which tunes the prior *geometric distribution* [5] over the time in the current state. (A range of geometric distributions are shown in Fig 2d.) There are many situations where this is a reasonable model, particularly if the observation frequency is close to the typical time in the state, but it is often desirable to have a model which allows specifying different temporal priors, particularly if the observation frequency is much higher than the typical time in state.

This can be achieved by using an additional probability distribution

$$P(t|\mathsf{s}){:=}P(\text{spent exactly } t \text{ timesteps in state} \mid \text{state is} \mathsf{s}) \quad (4)$$

instead of adding the self-transitions. These are assumed to be full distributions with non-zero (but increasingly small) probabilities for arbitrarily large durations. The probability of a sequence of observations is then composed from the probabilities of the transitions, state durations and features. This model is called a *Hidden Semi-Markov Model (HSMM)* (also known as an *explicit duration HMM*) [8, 17] where the equivalent of Eq 2 is the joint probability of the observations and the division of the sequence into intervals in a given state, ie, s_0 during $\tau_0{:}\tau_1, \ldots, \mathsf{s}_m$ during $\tau_m{:}\tau_{m+1}$,

$$P(\mathbf{f}_{0:T}, \mathsf{s}_0 \text{ during } \tau_0{:}\tau_1, \ldots, \mathsf{s}_m \text{ during } \tau_m{:}\tau_{m+1})$$

$$= \left[\prod_{i=1}^{m} P(\mathsf{s}_i|\mathsf{s}_{i-1})\right]\left[\prod_{i=0}^{m} P(\mathbf{f}_{\tau_i:\tau_{i+1}}|\mathsf{s}_i)\right]\left[\prod_{i=0}^{m} P(\tau_{i+1}{-}\tau_i|\mathsf{s}_i)\right] \quad (5)$$

$$= P(\mathbf{f}_{\tau_0:\tau_1}|\mathsf{s}_0)P(\tau_1{-}\tau_0|\mathsf{s}_0) \quad (6)$$

$$\times \left[\prod_{i=1}^{m} P(\mathsf{s}_{i+1}|\mathsf{s}_i)P(\mathbf{f}_{\tau_i:\tau_{i+1}}|\mathsf{s}_i)P(\tau_{i+1}{-}\tau_i|\mathsf{s}_i)\right]$$

$$= P(\mathbf{f}_{\tau_0:\tau_m}|\mathsf{s}_0 \text{ during } \tau_0{:}\tau_1, \ldots, \mathsf{s}_{m-1} \text{ during } \tau_{m-1}{:}\tau_m)$$

$$\times P(\mathsf{s}_m|\mathsf{s}_{m-1})P(\mathbf{f}_{\tau_m:\tau_{m+1}}|\mathsf{s}_m)P(\tau_{m+1}{-}\tau_m|\mathsf{s}_m) \quad (7)$$

where "s_m during $\tau_m{:}\tau_{m+1}$" means the system was in state s_m from time τ_m to $\tau_{m+1} - 1$ and we require $\tau_0{=}1$ and $\tau_{m+1}{=}T$. Again, Eq 5 expresses the probability in terms of the feature, duration and transition components taken separately. (There is an assumption here that the time-in-state and observed features are independent given the state, which is often reasonable.) Similarly in Eq 6 & Eq 7 they have been shuffled to show the Markov structure, although now *present* means 'current segment'.

Models for state duration. The top row of Fig 2 shows a set of histograms of state duration for various classes obtained from the manually labelled ground truth (discussed

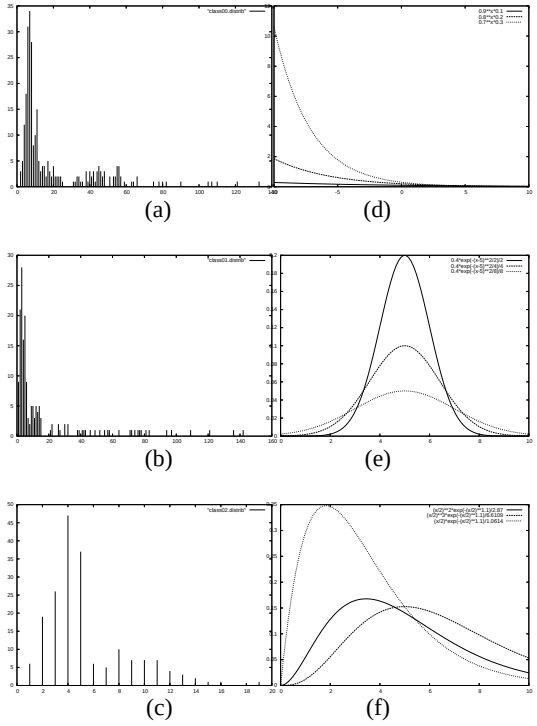


Figure 2: (left) histograms of time-in-state; geometric; (right) geometric, gaussian & modified gamma distributions

below). These have a rough characteristic shape where the probability of a given duration rises quickly to a most likely duration, then tails off significantly more slowly, a poor fit with the set of shapes obtained with a geometric distribution.

The obvious approach is to model the duration distribution using an empirical histogram. However, because each sample comes from a segment comprising perhaps tens of frames obtaining sufficient training data to adequately populate the histogram is difficult. Although we could try histogram smoothing, we choose instead to model the distribution with a parametric distribution. Two advantages are that we automatically generate a smooth distribution from the data and can accommodate prior knowledge about the likely time-in-state via priors on the parameters. We argue that using a roughly appropriately shaped parametric distribution is acceptable when training data is limited, even though we don't know if the 'true' generating distribution is in the chosen family. (An argument could be made for possible multi-modality in these distributions, which might suggest a mixture model, but with at most 100–200 examples per class – from a 12,500 frame ground truth – a meaningful conclusion cannot be drawn about this.)

To choose the parametric family, observe that the time-in-state is restricted to positive values and the empirical distributions in Fig 2 are noticeably asymmetric around the mode. Consequently a gaussian is also a poor model and instead we model the time-in-state with a 'modified gamma distribution'. This generalisation of the gamma distribution [5] parameterised by γ and β has pdf

$$p(t) = (t/\beta)^{\gamma} \exp(-(t/\beta)^{z})/Z \tag{8}$$

where z is some constant >1 and normalizing constant Z is equal to $\beta\Gamma(\frac{\gamma+1}{z})/z$. (As we use different distribution parameters for each state s the normalising constants *do not* cancel.) The plots in Fig 2 show examples of geometric, gaussian and modified gamma distributions; it is clear that the modified gamma is the most similar to the empirical plots.

Fitting parameters given training data is complicated since even with 100–200 examples there are still areas of low sample density and wide variation between adjacent time points. Although we attempted to fit distributions automatically, the restricted number of samples resulted in curves distorted by the requirements of fitting against the inadequately sampled right-hand tails of the histograms; consequently we fitted appropriate looking distributions manually for our experiments. More extensive ground truth datasets would enable automatic fitting of the distributions.

3. Computing the most likely HSMM decoding of a sequence

The activity recognition process is formulated as finding the most likely sequence of states according to the HSMM parameters. An $O(S^2T^2)$ HSMM most-likely sequence inference algorithm which works with arbitrary duration distributions is given in [17]. This time complexity essentially arises because at each timestep a maximisation has to be performed not only over all states for the previous timestep but also all possible lengths of time in that state. This quadratic behaviour is infeasible when working with long video sequences (large T), so we develop an $O(S^2T)$ algorithm (adapting [11]) by first showing the key step in inference is deciding between different segmentations ending at the same time, then show how this can be tackled effectively using structure within the HSMM model.

HSMM inference as optimal segmentation. For the standard HMM the natural view of inference is that of *assigning* states at each time, whereas for the HSMM inference is most naturally viewed as *segmenting* the sequence into intervals with a common state. We want to find the best division into segments using an algorithm which is *recursive* in that the optimal segmentation of a sequence can be computed using the optimal divisions of its prefixes. In a similar way to the standard HMM algorithm at each time t we find, for each state s, the best segmentation which ends in state s at t. The Markov property of the HSMM means that the

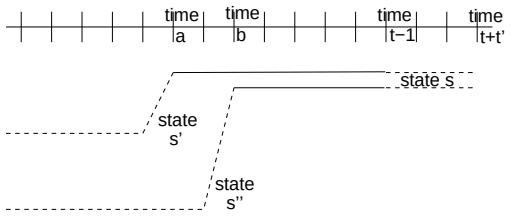


Figure 3: two potential segmentations, both of which transition to a final the same final state s

only influence from the previous segments comes from the total probability of that previous state and the time at which the transition to the present state occurred. This is illustrated by Fig 3, where the top line indicates the timeline in state s, whilst the other two lines show two segmentations which switch to s for the final block. The dashed segments on the left indicate the best probability previous segmentations (where we will see below the exact details of which don't matter). So the basic task is to decide between two such segmentations, both of which finish in a segment in state s at time $t-1$.

Structure in HSMM probabilities. We will show that, with a standard 'naive Bayes' assumption on features $P(\mathbf{f}_{a:b}|\mathsf{s}) = \prod_{i=a}^{b-1} P(\mathbf{f}_i|\mathsf{s})$, which of two segmentations ending in s is best *doesn't depend on the feature data in their common suffix*.

To do this we compare the probabilities of two hypothesised segmentations which have the common feature of both finishing at time $t-1$ in a segment in state s. (A schematic illustration shown in Fig 3.) In one case this final segment begins at time a with and has a probability $P_{h_1}=P(\mathbf{f}_{1:a}|to\_a)$ for a hypothesised segmentation to_a for the initial sequence $\mathbf{f}_{1:a}$. (Fig 3 shows the previous history before the transition being in a state s$'$ but the key point is that the only factor affecting the calculations is the probability $P(\mathbf{f}_{1:a}|to\_a)$). We have a second sequence which transitions to s at time b (where $a<b$) and has hypothesised segmentation to_b for $\mathbf{f}_{1:b}$ with probability$P_{h_2}=P(\mathbf{f}_{1:b}|to_b)$. Then the full segmentations including the final segment have probabilities

$$\begin{aligned} P'_{h_1} &:= P(\mathbf{f}_{1:t}|to\_a, \mathsf{s} \text{ during } a{:}t) && (9)\\ &= P(\mathbf{f}_{1:a}|to\_a)P(\mathbf{f}_a|\mathsf{s})\cdots P(\mathbf{f}_b|\mathsf{s})\cdots P(\mathbf{f}_t|\mathsf{s})P(t-a|\mathsf{s}) \\ &= P_{h_1}P(\mathbf{f}_a|\mathsf{s})\cdots P(\mathbf{f}_b|\mathsf{s})\cdots P(\mathbf{f}_t|\mathsf{s})P(t-a|\mathsf{s}) \\ P'_{h_2} &:= P(\mathbf{f}_{1:t}|to\_b, \mathsf{s} \text{ during } b{:}t) && (10)\\ &= P_{h_2}P(\mathbf{f}_b|\mathsf{s})\cdots P(\mathbf{f}_t|\mathsf{s})P(t-b|\mathsf{s}) \end{aligned}$$

If our sequence ended at time t we would choose the larger of P'_{h_1} and P'_{h_2} as the most likely sequence. Notice that both expressions have common terms Eq 9 and 10; defining

$$K := P(\mathsf{s}|\mathbf{f}_b)\cdots P(\mathbf{f}_t|\mathsf{s}) \quad (11)$$

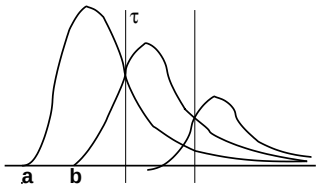


Figure 4: how curves of 'constant×duration' prior split the future into intervals

$$\begin{aligned} C_1 &:= P_{h_1}P(\mathbf{f}_a|\mathsf{s})\cdots P(\mathbf{f}_{b-1}|\mathsf{s}) && (12)\\ C_2 &:= P_{h_2} && (13) \end{aligned}$$

the rhs of equations 9 & 10 become $C_1KP(t-a|\mathsf{s})$ & $P'_{h_2}:=C_2KP(t-b|\mathsf{s})$ respectively, so that K is the probability of observations falling within the common suffix of the final segments in state s in Fig 3). The key is to now consider what happens if we extend the final segment of both possibilities (shown by the dashed lines on the right in Fig 3), remaining in the same state s, so they now end at a time $t+t'$:

$$\begin{aligned} & P(\mathbf{f}_{1:t+t'}|to\_a, \mathsf{s} \text{ during } a{:}t+t') && (14)\\ =\ & C_1KP(\mathbf{f}_{t+1}|\mathsf{s})\cdots P(\mathbf{f}_{t+t'}|\mathsf{s})P(t-a+t'|\mathsf{s}) \\ =\ & C_1KK'P(t-a+t'|\mathsf{s}) && (15) \end{aligned}$$

where we define $K'{:=}P(\mathbf{f}_{t+1}|\mathsf{s})\cdots P(\mathbf{f}_{t+t'}|\mathsf{s})$. Likewise

$$\begin{aligned} & P(\mathbf{f}_{1:t+t'}|to\_b, \mathsf{s} \text{ during } b{:}t+t') && (16)\\ =\ & C_2KP(\mathbf{f}_{t+1}|\mathsf{s})\cdots P(\mathbf{f}_{t+t'}|\mathsf{s})P(t-b+t'|\mathsf{s}) \\ =\ & C_2KK'P(t-b+t'|\mathsf{s}) && (17) \end{aligned}$$

We see the additional feature-dependent terms are again common to both expressions. The inference algorithms will proceed through the sequence finding the largest probability for the various combinations of final segment length and label. From these equations we can see that, in determining which of Eq 14 & 16 is larger we can cancel the common term KK' (regardless of its precise value) and thus need only compare $C_1P(t-a+t'|\mathsf{s})$ with $C_2P(t-b+t'|\mathsf{s})$. Thus we have a slightly simpler optimisation problem, but we can really exploit this if we additionally require that the duration distribution $P(t|\mathsf{s})$ is *concave monotonic*, i.e.,

$$\begin{aligned} & C_1P(t-a|\mathsf{s}) \le C_2P(t-b|\mathsf{s}) \\ \Rightarrow\ & C_1P(t-a+t'|\mathsf{s}) \le C_2P(t-b+t'|\mathsf{s}) && (18) \end{aligned}$$

for $t'>0$ and arbitrary constants C_1 and C_2. This intuitively states that 'given two segmentations ending in the same state at a common time, once they have been extended with new frames' data until the longer segmentation has a lower probability, then it will *always* have a lower probability as both segmentations are further extended over the feature sequence'. If we keep track of candidate segmentations and

initialisation

for $\mathsf{s}\in\mathcal{S}$: *#initialise each queue with one big interval*
  set $q[\mathsf{s}]\coloneqq\langle r\rangle$ with $r.nlp\coloneqq 0$, $r.t_0\coloneqq 1$ & $r.dom\_int\_st\coloneqq 1$

main algorithm

for $t\coloneqq 1,\ldots,T$: *#build up the best solution recursively*
  for $\mathsf{s}\in\mathcal{S}$ such that $t\geq q[\mathsf{s}].hd.next.dom\_int\_st$:
    $pop\_head(q[\mathsf{s}])$ *#discard intervals as time passes domination point*
  for $\mathsf{s}\in\mathcal{S}$: *#figure what'd happen if sequence ends in state* s
    create r with $r.nlp\coloneqq\infty$ *#make* r *'impossibly unlikely'*
    for $\mathsf{s}'\in\mathcal{S}\backslash\{\mathsf{s}\}$: *#find best transition from other state into* s *at time* t
      add $-\log p(\mathbf{f}_t|\mathsf{s}')$ to $.nlp$ of each entry on $q[\mathsf{s}']$
      $q[\mathsf{s}'].hd.nlp\coloneqq$sum of old nlp, duration & transition values
      if $q[\mathsf{s}'].hd.nlp<r.nlp$: $r\coloneqq q[\mathsf{s}'].hd$ *#best new 'solution' yet*
    loop: *#discard intervals off queue until get split interval*
      $\tau\coloneqq dom\_pt(q[\mathsf{s}].tl.nlp, q[\mathsf{s}].tl.t_0, r.nlp, r.t_0)$
      if $\tau\geq q[\mathsf{s}].tl.dom\_int\_st$:*#*$r$ *dominates* $q[\mathsf{s}].tl$ *from* τ *onwards*
        exit loop *#almost finished updating dominant intervals*
      $pop\_tail(q[\mathsf{s}])$ *#solution* $q[\mathsf{s}].tl$ *never optimal, so discard*
    set r' with $r'.nlp\coloneqq r.nlp$, $r'.t_0\coloneqq t$,& $r'.dom\_int\_st\coloneqq\tau$
    $push\_tail(q[\mathsf{s}], r')$ *#newly created interval occurs at end*

finding the final result

search over final states s to find the $q[\mathsf{s}].hd$ with lowest $.nlp$

Records describe dominant intervals via fields $.nlp$ *(solution's negated log-probability so far),* $.t_0$ *(timestep the final segment begins) &* $.dom\_int\_st$ *(time current interval becomes dominant). The* $q[\mathsf{s}]$*'s are double ended queues with head* hd *and tail* tl*.*

Figure 5: Overview of $O(S^2T)$ HSMM decoding algorithm

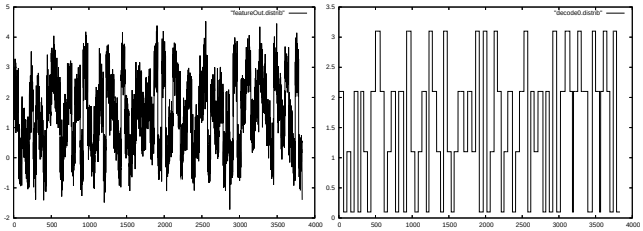
their probabilities we can detect the point at which Eq 18 becomes valid and then deduce some segementations which can *never* give optimal solutions in the future *regardless of the sequence's stopping point or the new feature data*; via this means most segmentations can be discarded. (We still need to compute the probabilities of those possibilities that we do keep using all the terms, but we can rule out some possibilities no matter what future feature values are observed using only the values C_1 and C_2 and the time t at which Eq 18 first becomes true.)

To relate this to the dynamic programming algorithms of [11, 7], we take negated log-probabilities, so Eq 18 becomes *convex monotonicity*:

$$\begin{aligned} & \log(C_2/C_1)+L_s(t-a) \geq L_s(t-b) \\ \Rightarrow\quad & \log(C_2/C_1)+L_s(t-a+t') \geq L_s(t-b+t') \end{aligned} \quad (19)$$

where $L_s(T)=-\log(T|\mathsf{s})$. Thus we can work in the log-probability domain, which removes most of the underflow problems that occur with large products of probabilities.

Convex monotonicity and dominant intervals. As our duration distributions have a particular parametric form, not



(a) (b)

sequence length	1920	2880	3840	4800	5760	6720	7680
# queue insertions	7620	11500	15320	19184	22980	26752	30660
# queue deletions	7613	11489	15305	19172	22970	26736	30654
# queue ops	15233	22989	30625	38356	45950	53488	61314
# bound $2ST$	15360	23040	30720	38400	46080	53760	61440

Figure 6: Feature output (graph a), HSMM decoded sequence (graph b), and $O(ST)$ linear scaling for the number of insert/delete operations for synthetic datasets

only can we verify algebraically that Eq 19 holds but also find the time at which the shorter solution (ie, most recent transition to_b in Fig 3) first becomes better, i.e.

$$dom\_pt(C_1,a,C_2,b)\coloneqq\tau \text{ st } \log(C_2/C_1)+L_s(\tau-a)\coloneqq L_s(\tau-b) \quad (20)$$

in terms of negated log-probabilities, seen schematically in 3. We call τ the *domination point* because the segmentation b 'dominates' segmentation a when the final segment is extended to *any* time after τ. For the distribution in Eq 8 the original probability based equivalent of Eq 20 for τ is

$$\begin{aligned} & C_1\left(\frac{\tau-a}{\beta}\right)^{\gamma}\exp\left[-\left(\frac{\tau-a}{\beta}\right)^{z}\right] \\ =& C_2\left(\frac{\tau-b}{\beta}\right)^{\gamma}\exp\left[-\left(\frac{\tau-b}{\beta}\right)^{z}\right] \end{aligned} \quad (21)$$

This doesn't have a closed form, but as L_s is monotonic we can use binary search to find the zero of Eq 20. (Histogram-based time-in-state distributions could be used in the same way *providing they satisfy concave monotonicity condition Eq 18*.)

To complete the algorithm recall that $\geq$ is *transitive*, so that if we're at a time t and have a segmentation α dominating β, then any segmentation γ dominated by β is thus automatically dominated by α. With a little thought we can see that for each state s we can divide the future into intervals by finding the domination points. Each interval describes the set of timesteps where a given segmentation is optimal, illustrated schematically in Fig 4b. (There will typically be only a small number of intervals where different labellings dominate because most segmentations are dominated by one solution or another for the entire future.) The key point is that these intervals do not need to be found from scratch at each new step but, due to convex monotonicity, can be updated by finding the domination point for the new segmentation against the previous timesteps' dominant intervals, starting from the final interval ending at $t=\infty$ and

Figure 7: Typical frames and discriminative image patches (fi nal row, different scale) for the breakfast TV sequence

discarding intervals until one is found where the domination point falls within its.

A detailed overview of the algorithm for fi nding the most probable sequence segmentation is shown in Fig 5, using queues of records describing intervals over which a given segmentation is dominant for pruning. To analyse its running time we start by noting the three nested **for** loops give $O(S^2T)$ time. To see the contribution of the queue maintenance **loop**, note: (i) each $q[\mathsf{s}]$ is initialised with 1 record in step 2; (ii) for each state s at time t one record r' is pushed onto the queue in step 18; (iii) each record can be popped off at most once during the entire run (at either step 5 or 16). Thus only $O(T)$ operations are performed on each $q[\mathsf{s}]$ and the run-time is governed by the $O(S^2T)$ double-nested **for** term. (The overview is simplifi ed: the **for** loops updating each queue $q[\mathsf{s}]$ should 'see' the non-updated heads $q[\mathsf{s}']$**.**hd of the other queues in step 7, which requires book-keeping; also real code has to be structured to avoid problems from loss of precision with machine floating point.)

3.1. Experiments

We demonstrate two points: Firstly, given HSMM parameters and an observation sequence our algorithm produces the optimal output in $O(S^2T)$ time. Secondly, that a HSMM is a *practical approximation* for recognising behaviours in real video sequences. The model does not use all the human labeller's knowledge, so we wouldn't expect the best Hidden Semi-Markov *Model* decoding to *exactly* match the ground truth. Such errors are defi ciencies of the HSMM assumptions; our algorithm fi nds the optimal solution given the model.

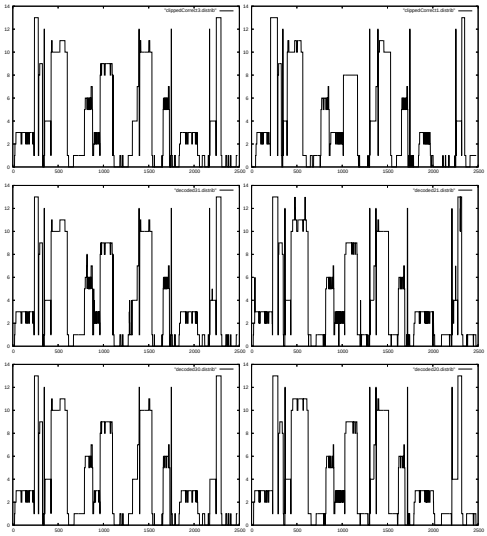
Synthetic data. Our experiment used a synthetic generating process with four states 0,..., 3 and modifi ed gamma distributions with random parameters for time-in-state. The observations are the state number plus additive gaussian noise. Graph a in Fig 6 shows the raw feature output and graph b shows the most likely HSMM decoding for a typical random dataset. For a large number of random runs we used Monte-Carlo methods around the found HSMM to sample similar solutions around the produced output state sequence and checked that each variation had a higher negated log-probability, thus verifying our algorithm found the correct (ie, most likely) solution. Finally the table in Fig 6 shows the good linear agreement between theoretical value $\approx ST$ and the number of insertions/deletions on the queues. (The difference #ins$-$#dels is the number of intervals on the various queues when the sequence ends.)

Real data. For our real-world experiments we used footage from a British breakfast television programme, since long sequences of $\approx 2^1/_4$ hours are easily obtainable and there is a strong structure to the broadcast. We took fi ve MPEG enocoded video sequences and extracted every 50th frame (i.e., 2 seconds). Fifteen labels were chosen and the appropriate transition structures found (shown in Fig 1). After hand-labelling we could then learn transition probabilities and duration distributions for both a standard HMM and a HSMM.

Low-level features were obtained by fi rst creating a training set of frames and repeatedly selecting large sets of random patches with locations in the images and computing Sum of Absolute Difference values with other frames in the set. Blocks were then assigned to the class for which they gave lowest score and a 1-dimensional kernel density estimate [2] for each class produced. On the test sequence these patches and models were then used to produce likelihoods $P(\mathbf{f}|\mathsf{s})$ of each image given the state label. The likelihoods were then combined to give probabilities of the image being in each class. Finally the state labels for the whole sequence was computed using the HMM and HSMM algorithms.

We compared our algorithm and the HMM model's match with the ground truth over each of the fi ve sequences, with the results for two sequences shown in 8. The top row shows the ground truth (again plotting the numeric state label at each time), the middle row the HMM decoding and the bottom row the HSMM labellings for two of the datasets. Some additional artifacts can be seen in the HMM decoding due to its inability to model durations. The table shows the percentage of correct labellings for each of the 5 sequences for both approaches. (The values are high because many of the correlation blocks correspond to highly distinctive information like on-screen graphics which provides a very strong data cue over signifi cant portions of the input sequence, thus diminishing the importance of the temporal structure.) Nevertheless, the HSMM has labelled a higher percentage of frames correctly. The practical runtimes of both algorithms were essentially the same.

The advantages of a HSMM over a HMM would greater if we extracted every frame from the footage, since two sec-



HMM correct	94.4	94.1	93.4	94.5	94.5
HSMM correct	97.4	98.2	97.2	97.5	97.4

Figure 8: Results graphs for the five test sequences

onds would give 50 observations rather than one.

4. Conclusions & future work

We reviewed the Hidden Markov Models and Hidden Semi-Markov Models and showed the duration-in-state distributions for the TV datasets is better well modelled by a standard HSMM model. This paper's key contribution is an algorithm for efficient inference in HSMMs with convex monotone duration distributions in the case of naive Bayes features and suitable duration distributions. In future we intend to apply these algorithms to the task of understanding surveillance footage with the context of the CAVIAR vision project [9].

Acknowledgements. This work was supported under EC Funded CAVIAR project/IST 2001 37540. The authors thank the reviewers for their comments.

References

[1] R E Bellman. *Dynamic Programming*. Princeton, NJ, 1957.

[2] C Bishop. *Neural Networks for Pattern Recognition*. OUP, 1995.

[3] M Brand. Understanding manipulation in video. In *Proc AFGR*, pages 96–99, 1996.

[4] M Brand, N Oliver, and A P Pentland. Coupled Hidden Markov Models for complex action recognition. In *Comp. Vis. and Pattern Recog.*, Los Alamitos, 1997.

[5] I Bronshtein, K Semendyayev, G Musiol, and H Muehlig. *Handbook of Mathematics*. Springer, 2003.

[6] J L Crowley and P Reignier. Dynamic composition of process federations for context aware perception of human activity. In *Int Conf on Integration of Knowledge Intensive Multi-Agent Systems*, 2003.

[7] D Eppstein. Sequence comparison with mixed convex and concave costs. *J. Algorithms*, pages 85–101, 1990.

[8] J Ferguson. Variable duration models for speech. In *Proc. Symp. on the Application of HMMs to Text and Speech*, pages 143–179, 1980.

[9] Fisher *et al*. homepages.inf.ed.ac.uk/rbf/ CAVIAR, 2004.

[10] A Galata, A Cohn, D Magee, and D Hogg. Modeling interaction using learnt qualitative spatio-temporal relations and variable length markov models. In *Proc Euro. Conf on Artificial Intelligence*, Lyon, 2002.

[11] Z Galil and K Park. Dynamic programming with convexity, concavity and sparsity. Technical report, CUCS-066-90, Dept Comp. Sci, Columbia Univ., 1990.

[12] Xianping Ge. *Segmental Semi-Markov Models and Applications to Sequence Analysis*. PhD thesis, Dept Comp Sci, Uni. California, Irvine, 2002.

[13] S Hongeng, R Nevatia, and F Bremond. Video-based event recognition: activity representation and probabilistic methods. *Comp. Vis. and Image Understanding*, 96(129–162), 2004.

[14] Y Ivanov and A Bobick. Recognition of visual activities and interactions by stochastic parsing. *PAMI*, 20(8):852–872, 2000.

[15] M Johnson. Capacity and complexity of HMM duration modelling techniques. *IEEE Sig Proc Letters*, 12(5), 2005.

[16] D Minnen, I Essa, and T Starner. Expectation grammars: Leveraging high-level expectations for activity recognition. In *IEEE Int Conf on CVPR*, Madison, 2003.

[17] C Mitchell, M Harper, and L Jamieson. On the complexity of Explicit Duration HMMs. *IEEE Trans on Speech and Audio Processing*, 3(3), 1995.

[18] N Oliver, E Horvitz, and A Garg. Layer representations for human activity recognition. *Computer Vision and Image Understanding*, (9):163–180, 2004.

[19] L Rabiner and B Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16, 1986.

[20] N Rea, R Dahyot, and A Kokaram. Semantic event detection in sports through motion understanding. In *Int Conf on Image and Video Retrieval*, Dublin, 2004.

[21] N Rota and M Thonnat. Video sequence interpretation for visual surveillance. In *Int Workshop on Visual Surveillance*, Dublin, 2000.

[22] D Phung T Duong, H Bui and S Venkatesh. Activitiy recognition and abnormality detection with the switching Hidden Semi-Markov Model. In *Int. Conf. on Comp. Vis. & Pat. Recog*, 2005.

[23] S-Z Yu and H Kobayashi. An efficient forward-backward algorithm for an explicit duration Hidden Markov Model. *IEEE Sig Proc Letters*, 10(1), 2003.