Optimising Dynamic Graphical Models for Video Content Analysis

Tao Xiang* and Shaogang Gong
Department of Computer Science
Queen Mary, University of London, London E1 4NS, UK
{txiang,sgg}@dcs.qmul.ac.uk

Abstract

A key problem in video content analysis using dynamic graphical models is to learn a suitable model structure given some observed visual data. We propose a Completed Likelihood AIC (CL-AIC) scoring function for solving the problem. CL-AIC differs from existing scoring functions in that it aims to optimise *explicitly* both the explanation and prediction capabilities of a model simultaneously. CL-AIC is derived as a general scoring function suitable for both static and dynamic graphical models with hidden variables. In particular, we formulate CL-AIC for determining the number of hidden states for a Hidden Markov Model (HMM) and the topology of a Dynamically Multi-Linked HMM (DML-HMM). The effectiveness of CL-AIC on learning the optimal structure of a dynamic graphical model especially given sparse and noisy visual date is shown through comparative experiments against existing scoring functions including Bayesian Information Criterion (BIC), Akaike's Information Criterion (AIC), Integrated Completed Likelihood (ICL), and Variational Bayesian (VB). We demonstrate that CL-AIC is superior to the other scoring functions in building dynamic graphical models for solving two challenging problems in video content analysis: (1) content based surveillance video segmentation, and (2) discovering causal/temporal relationships among visual events for group activity modelling.

Keywords: Video content analysis, structure scoring, graphical models, hidden Markov models, surveillance video segmentation, group activity modelling.

1 Introduction

Dynamic graphical models or Dynamic Bayesian Networks (DBNs), especially Hidden Markov Models (HMMs) and their variants, have become increasingly popular for modelling and analysing dynamic video

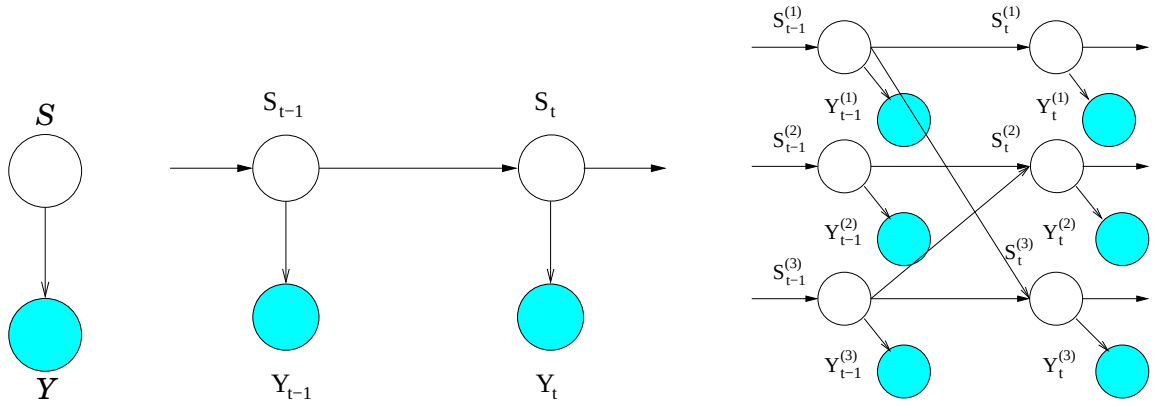
*Corresponding author. Tel: (+44)-(0)20-7882-5201; Fax: (+44)-(0)20-8980-6533

content [13, 21, 8, 12, 26, 19, 15, 33]. By using a DBN for video content analysis, we assume that dynamic visual content is generated sequentially by some hidden states of the dynamic scene which evolve over time. These hidden states often have physical meanings. For instance, they could correspond to certain stages/phases of an activity [12, 26, 21], the occurrence of different classes of visual events [19], or different types of transition segments between video shots [8]. The hidden states, as suggested by the name, cannot be observed directly. They can only be inferred from the observed visual data given a learned DBN. Learning a DBN involves estimating both its structure and parameters from data. The structure of a DBN refers primarily to (1) the number of hidden states of each hidden variables of a model and (2) the conditional independence structure of a model, i.e, factorisation of the state space for determining the topology of a graph. There have been extensive studies in the machine learning community on efficient parameter learning when the structure of the model is known *a priori* (i.e. assumed) [18]. However, much less effort has been made to tackle the more challenging problem of learning the optimal structure of an unknown DBN [4, 17, 11, 35]. Most previous DBNs-based video content modelling approaches avoid the structure learning problem by setting the structure manually [21, 26, 8, 15]. However, it has been shown that a learned structure can be advantageous over manually set ones given sparse and noisy visual data [19]. In this paper, we address the problem of how to accurately and robustly learn the optimal structure of a DBN for video content analysis in a realistic situation where only sparse and noisy visual data are available.

Most previous structure learning techniques have adopted a search-and-score paradigm [17] [1]. These techniques first define a scoring function/model selection criterion consisting of a maximum likelihood term and a penalty term to penalise complex models. The model structure space is then searched to find the optimal model structure with the highest score. The most commonly used scoring functions include Bayesian Information Criterion (BIC) [29] , Minimum Description Length (MDL) [28], BDe [20], Akaike's Information Criterion (AIC) [1], Integrated Completed Likelihood (ICL) [6], and Variational Bayesian (VB) [4, 5]. The selected models are 'optimal' in a sense that they can either best explain the existing data (BIC, MDL), or best predict unseen data (AIC). It has been demonstrated both theoretically and experimentally in the case of static models that explanation oriented scoring functions suffer from model under-fitting while prediction oriented ones suffer from model over-fitting [23, 30, 6, 34].

To address the problems associated with existing scoring functions, we argue that a better scoring function should select a model structure that is capable of both explaining the observed data and predicting unseen data optimally at the same time. To this end, we derive Completed Likelihood AIC (CL-AIC) for

[1] Alternatives include the Bayesian approach to model selection [4] and context-specific independence [9].



(a) GMM (b) HMM (c) DML-HMM

Figure 1: Three different types of graphical models with hidden nodes, among which HMM and DML-HMM are DBNs. Observation nodes are shown as shaded circles and hidden nodes as clear circles.

learning the structure of a DBN. CL-AIC was first introduced in our previous work [34] for Gaussian Mixture Models (GMMs) which can be represented as a static graphical model (see Fig. 1(a)). In this paper, CL-AIC is derived as a general scoring function suitable for both static and dynamic graphical models, with GMMs and DBNs as special cases. In particular, CL-AIC is formulated for determining the number of hidden states of a HMM and for learning the topology of a Dynamically Multi-Linked HMM (DML-HMM) (see Fig. 1(b)&(c)).

The effectiveness of CL-AIC on DBNs structure learning is demonstrated through comparative experiments against Bayesian Information Criterion (BIC), Akaike's Information Criterion (AIC), Integrated Completed Likelihood (ICL), and Variational Bayesian (VB). Experiments on synthetic data were carried out to examine and quantify the effect of sample size on the performance of different score functions. The results, for the first time, reveal a key difference in structure learning of static and dynamic graphical models in terms of the definition of data sparseness. We further considered two video content analysis problems using real data: (1) content based surveillance video segmentation, and (2) discovering causal/temporal relationships among visual events for group activity modelling. Our experimental results demonstrate that our CL-AIC is superior to alternative scoring functions in building dynamic graphical models for video content analysis especially given sparse and noisy data.

The rest of the paper is structured as follows: in Section 2, we derive CL-AIC as a general scoring function for graphical models with hidden variables. We also formulate CL-AIC for two special cases of DBN, namely a HMM and a DML-HMM, and present synthetic experiments to compare CL-AIC to existing scoring functions including BIC, AIC, ICL, and VB. In Section 3, we address the problem of learning the optimal number of video segments for surveillance video segmentation. Comparative experiments are

conducted using over 10 hours of challenging outdoor surveillance video footages. We then compare CL-AIC with other competing scoring functions in learning the topology of a DML-HMM for group activity modelling in Section 4. The paper concludes in Section 5.

2 Completed Likelihood AIC for Graphical Models with Hidden Variables

We derive Completed Likelihood AIC (CL-AIC) for graphical models with hidden variables with GMMs and DBNs as special cases. Let us first consider the nature of computation in estimating and using a graphical model. Consider an observed data set $\mathcal{Y}$ modelled by a graphical model $\mathcal{M}_K$ with hidden variables. $\mathcal{M}_K$ can be used to perform three tasks: (1) estimating the unknown distribution that most likely generates $\mathcal{Y}$, (2) inferring the values of hidden variable in $\mathcal{M}_K$ from $\mathcal{Y}$, and (3) predicting unseen data. Computing (1) and (2) emphasises data explanation while solving (3) concerns with data prediction and synthesising. In this context, scoring functions based on approximating the Bayesian Model Selection criterion [27] such as BIC and Variational Bayesian (VB) choose a model that maximises $p(\mathcal{Y}|\mathcal{M}_K)$, the probability of observing $\mathcal{Y}$ given $\mathcal{M}_K$ or the marginal likelihood of the model. They thus enforce mainly task (1). AIC, on the other hand, chooses the model that best predicts unseen data, therefore optimising (3). CL-AIC utilises Completed Likelihood (CL) in order to makes explicit the task (2), while following a derivation procedure similar to that of AIC.

Completed Likelihood (CL) was originally derived for as a scoring function for mixture models [7, 6]. The model that best explains both the observed data and hidden variables inferred from the observed data is selected by CL as the optimal model. CL is thus a strongly explanation-oriented scoring function. Here, we first extend the definition of Completed Likelihood (CL) to a general case of graphical models with hidden variables. The complete data, denoted as $\bar{\mathcal{Y}}$, for such a model is a combination of the observed data ($\mathcal{Y}$) and the values of the hidden variables ($\mathcal{Z}$): $\bar{\mathcal{Y}} = \{\mathcal{Y}, \mathcal{Z}\}$, where $\mathcal{Z}$ is unknown, and must be inferred from $\mathcal{Y}$. The completed log-likelihood of $\bar{\mathcal{Y}}$ is:

$$\mathrm{CL}(K) = \log p(\mathcal{Y}|\mathcal{M}_K, \boldsymbol{\theta}_{\mathcal{M}_K}) + \log p(\mathcal{Z}|\mathcal{Y}, \mathcal{M}_K, \boldsymbol{\theta}_{\mathcal{M}_K})$$

where $\boldsymbol{\theta}_{\mathcal{M}_K}$ are the true model parameters and K is the index of the candidate models. In practice, $\boldsymbol{\theta}_{\mathcal{M}_K}$ are replaced using the Maximum Likelihood (ML) estimate $\hat{\boldsymbol{\theta}}_{\mathcal{M}_K}$ and the unknown values of the hidden variables $\mathcal{Z}$ is replaced by $\hat{\mathcal{Z}}$, the values inferred from the observed data $\mathcal{Y}$. The completed log-likelihood

is thus rewritten as:

$$\mathrm{CL}(K) = \log p(\mathcal{Y}|\mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K}) + \log p(\hat{\mathcal{Z}}|\mathcal{Y}, \mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K}) \tag{1}$$

CL-AIC aims to choose a model that best explains the the observed data and has the minimal divergence to the true model, therefore best predicting the unseen data as well. The divergence between a candidate model and the true model is measured using the Kullback-Leibler information [24]. Given a completed data set $\bar{\mathcal{Y}}$, we assume that $\bar{\mathcal{Y}}$ is generated by the unknown true model $\mathcal{M}_0$ with model parameter $\boldsymbol{\theta}_{\mathcal{M}_0}$. For any given model $\mathcal{M}_K$ and the Maximum Likelihood Estimate $\hat{\boldsymbol{\theta}}_{\mathcal{M}_K}$, the Kullback-Leibler divergence between the two models is computed as

$$d(\mathcal{M}_0, \mathcal{M}_K) = E\left[\log\left(\frac{p(\bar{\mathcal{Y}}|\mathcal{M}_0, \boldsymbol{\theta}_{\mathcal{M}_0})}{p(\bar{\mathcal{Y}}|\mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K})}\right)\right]. \tag{2}$$

Ranking the candidate models according to $d(\mathcal{M}_0, \mathcal{M}_K)$ is equivalent to ranking them according to

$$\delta(\mathcal{M}_0, \mathcal{M}_K) = E\left[-2\log p(\bar{\mathcal{Y}}|\mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K})\right].$$

$\delta(\mathcal{M}_0, \mathcal{M}_K)$ cannot be computed directly since the unknown true model is required. However, it was noted by Akaike [1] that $-2\log p(\bar{\mathcal{Y}}|\mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K})$ can serve as a biased approximation of $\delta(\mathcal{M}_0, \mathcal{M}_K)$, and the bias adjustment

$$E\left[\delta(\mathcal{M}_0, \mathcal{M}_K) + 2\log p(\bar{\mathcal{Y}}|\mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K})\right]$$

converges to $2C_K$ when the number of data sample approaches infinity. Our CL-AIC is thus derived as:

$$\text{CL–AIC(K)} = -\log p(\bar{\mathcal{Y}}|\mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K}) + C_K. \tag{3}$$

where C_K is the dimensionality of the parameter space of model $\mathcal{M}_K$. The first term on the right hand side of (3) is the completed likelihood given by Eqn. (1). We thus have:

$$\text{CL–AIC(K)} = -\log p(\mathcal{Y}|\mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K}) - \log p(\hat{\mathcal{Z}}|\mathcal{Y}, \mathcal{M}_K, \hat{\boldsymbol{\theta}}_{\mathcal{M}_K}) + C_K \tag{4}$$

The candidate model structure that has the minimal CL-AIC value will be chosen as the optimal structure.

By utilising an explanation-oriented scoring function (CL) and following a derivation procedure sim-

ilar to that of a prediction-oriented scoring function (AIC), CL-AIC is able to select the model structure that maximises both the probability of the observed data and the posterior probability of the inferred hidden/incomplete data simultaneously whilst having a minimum number of free parameters (i.e. as simple as possible). This formulation results in a number of important differences compared to existing techniques: (1) Unlike previous scoring functions, CL-AIC attempts to optimise *explicitly* the explanation and prediction capabilities of a model. This makes CL-AIC theoretically attractive. The effectiveness of CL-AIC in practice is demonstrated through experiments in the following sections. (2) It has been shown that Completed Likelihood can be combined with BIC which leads to an Integrated Completed Likelihood (ICL) scoring function [6]. However, the experiments reported in [6] indicated that in the case of mixture models, ICL performs poorly when data belonging to different mixture components are severely overlapped. We suggest this is caused by the factor that ICL is a combination of two explanation-oriented scoring functions without considering the prediction capability of the model. Since CL-AIC integrates an explanation-oriented scoring function with a prediction-oriented one, it is theoretically better justified than ICL. Our experiments in the following reinforce this observation.

As shown above, CL-AIC is derived based on a very intuitive principle and can be formulated for a general category of models, namely graphical models with hidden variables. Example of these models include both static statistical models such as GMMs, Naive Bayesian Classifiers, Principal Component Analysis (PCA), and Independent Components Analysis (ICA), and dynamic ones such as HMMs and their variants, and Kalman Filters. Since we are interested in Dynamic Bayesian Networks (DBNs) for video content analysis in this paper, we formulate CL-AIC for two cases of DBNs in the following.

2.1 Determining the Number of Hidden States of a HMM

Let us now consider a specific problem of learning the structure of a Hidden Markov Model (HMM). A HMM has a fixed topology with one hidden variable and one observation variable at each time instance t (see Fig. 1(b)). The hidden variable is discrete in most applications. The structure learning problem for a HMM thus refers to how to determine the number of hidden states that the hidden variable can assume. Assuming that at each time instance t, the discrete hidden variable S_t can have K different values (states), the complete data for the model is $\bar{\mathcal{Y}} = \{\mathcal{Y}, \mathcal{Z}\}$ where $\mathcal{Z}$ is the true hidden variable values over different

time instances (i.e. the true hidden state sequence). The completed log-likelihood of $\bar{\mathcal{Y}}$ is computed as:

$$\mathrm{CL}(K) = \log\left(\sum_{S} p(\mathcal{Y}|S, \hat{\boldsymbol{\theta}}_K) p(S|\hat{\boldsymbol{\theta}}_K)\right) + \log p(S = \hat{\mathcal{Z}}|\mathcal{Y}, \hat{\boldsymbol{\theta}}_K). \tag{5}$$

where $S = \{S_1, ..., S_T\}$ represents all the possible hidden state sequences, T is the length of the sequence, $\hat{\boldsymbol{\theta}}_K$ are the ML (maximum likelihood) estimate of the model parameters of a HMM with K hidden states, $\hat{\mathcal{Z}}$ is the most probable state sequence (i.e. the hidden state sequence among S that best explains the observation sequence) given $\hat{\boldsymbol{\theta}}_K$ and $\mathcal{Y}$. $\hat{\boldsymbol{\theta}}_K$ can be computed using the forward-backward (Baum-Welch) algorithm [3] based on dynamic programming and $\hat{\mathcal{Z}}$ can be obtained using the Viterbi algorithm [16]. We thus have:

$$\text{CL–AIC}(K) = -\log\left(\sum_{S} p(\mathcal{Y}|S, \hat{\boldsymbol{\theta}}_K) p(S|\hat{\boldsymbol{\theta}}_K)\right) - \log p(S = \hat{\mathcal{Z}}|\mathcal{Y}, \hat{\boldsymbol{\theta}}_K) + C_K. \tag{6}$$

A detailed description on how to use the forward-backward procedure and the Viterbi algorithm to efficiently compute the first and second terms on the right hand side of Eqn. (6) can be found at Rabiner's excellent HMM tutorial [25]. The optimal number of hidden states $\hat{K}$ is then determined as:

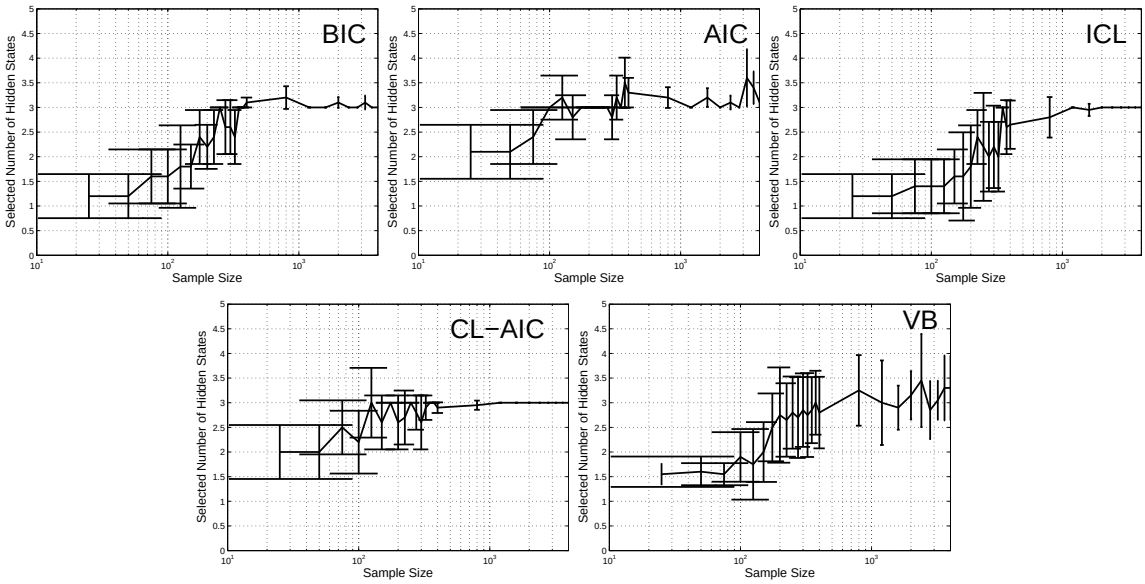
$$\hat{K} = \arg\min_{K} \text{CL–AIC}(K) \tag{7}$$

Synthetic experiments were conducted to compare the effectiveness of CL-AIC with that of Bayesian Information Criterion (BIC), Akaike's Information Criterion (AIC), Integrated Completed Likelihood (ICL), and Variational Bayesian (VB) on determining the number of hidden states of a HMM given data of different sample sizes. One-dimensional data were first generated from a 3-state HMM (i.e. the hidden variable at each time instance can assume 3 states) whose parameters are:

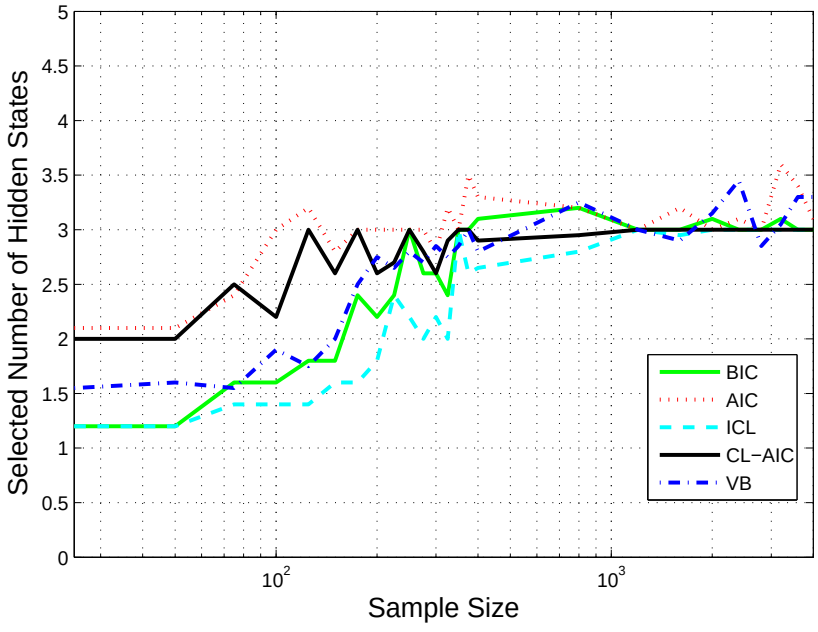
$$\mathbf{A} = \begin{bmatrix} 1/3 & 1/6 & 1/2 \\ 0 & 1/3 & 2/3 \\ 1/2 & 1/2 & 0 \end{bmatrix}, \pi = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}, \mathbf{B} = \left\{ \begin{array}{l} \mu_1 = 1, \sigma_1^2 = 0.5 \\ \mu_2 = 3, \sigma_2^2 = 0.5 \\ \mu_3 = 5, \sigma_3^2 = 0.5 \end{array} \right\}, \tag{8}$$

where $\mathbf{A}$ is the transition matrix, π is the initial state probability and $\mathbf{B}$ contains the parameters of the emission density (Gaussians with the indicated means and variances). The total number of free parameters is 14 for this HMM. The data were then perturbed by uniformly distributed random noise with a range of $[-0.5\ \ 0.5]$. HMMs with the number of hidden states K varying from 1 to 10 were evaluated. The five different scoring functions were tested on the data sets with the sample size T varying from 25 to 4000. To

avoid being trapped at local minima, the EM algorithm[2] used for estimating model parameters was randomly initialised for 20 times and the solution that yielded the largest likelihood after 30 iterations were chosen. The results are shown in Fig. 2 using the mean and ± 1 standard deviation of the selected number of hidden states over 50 trials, with each trial having a different random number seeds.



(a)

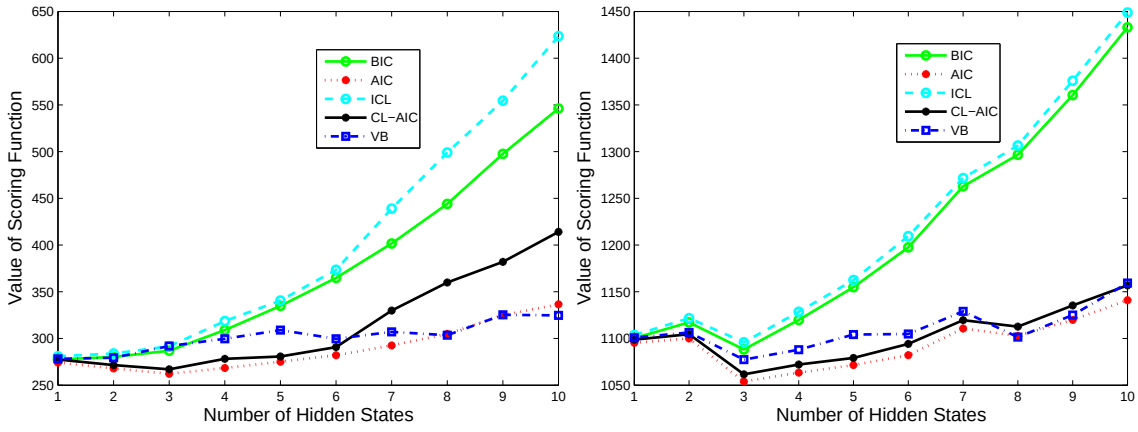


(b)

Figure 2: Synthetic data experiment results for determining the number of hidden states of a HMM using different scoring functions. (a) Selected number of hidden states (mean and ± 1 standard deviation over 50 trials); (b) Mean of the selected number of hidden states (The true number of hidden states is 3).

Fig. 2(b) shows the mean of the number of states estimated by different scoring functions over 50 trials in a single plot. It can be seen that when the sample sizes were small, all five scoring functions

[2] A modified EM algorithm was used for computing the Variational Bayesian (VB) scoring function [5].



(a) Sample size 250

(b) Sample size 1000

Figure 3: Examples of choosing optimal number of hidden states using different scoring functions. The value of 5 different scoring functions are plotted against the number of hidden states. The global minimals on the plots correspond to the selected numbers of hidden states. In (a), both AIC and CL-AIC chose the correct hidden state number 3 while the other three select 2 given a data set of 250 samples. In (b) all 5 scoring functions selected the correct number given a data set of 1000 samples.

tended to favour under-fitted models (i.e. underestimate the number of states), with AIC and CL-AIC clearly outperforming BIC, ICL and VB. As the sample sizes increased, the number of hidden states determined using all scoring functions converged to the true number 3. Given densely sampled data sets ($T > 400$), our results show that both AIC, BIC and VB tended to slightly over-fit while ICL and CL-AIC yielded more accurate estimation of the number of hidden states. Fig. 2(a) shows the variations of the structure learning results across different trials which indicate the sensitivity of different scoring function to the distribution of data and more importantly the presence of noise. It can be seen from Fig. 2(a) that both AIC and VB exhibited large variations in the estimated number of states no matter what the sample size was, while other scoring functions had smaller variations given larger sample sizes. Fig. 3 shows two examples of determining the optimal number of hidden states using different scoring functions. The value of different scoring functions is plotted against the number of hidden states in the candidate models. It can be seen that the scoring function value plots of AIC and VB have more local minimals and are less discriminative for different candidate model structures than the other three. This partially explains why the structure learning results for AIC and VB exhibited larger variations across different trials. Overall, the results show that CL-AIC has the best performance among the five scoring functions that we compared.

It can be seen from Fig. 2 and 3 that different scoring function perform differently as the sample size increases. In particular, Fig. 2(b) shows that when the data set is sparse, all five scoring functions tend to underfit the model. In the experiment presented in Fig. 2 and 3, the distribution of the observed data follows three Gaussians (see Eqn. (8)) and each hidden state will correspond to one Gaussian when both

the model structure and model parameters are learned accurately. When the sample size is small with the presence of noise, some of the three Gaussians may not be well supported by the data. This explains the underfitting tendency of the five scoring functions. Both Fig. 2(b) and Fig. 3(a) also show that given a sparsely sampled data set, CL-AIC and AIC outperform the other three. It is easy to understand why CL-AIC is less underfitting than ICL in this case. A comparison of the two shows that they only differ in their penalty terms in formulation. In particular, CL-AIC has a weaker penalty term, therefore always favouring more complicated model compared to ICL. For exact the same reason, AIC is less underfitting than BIC in Fig. 2(b) and Fig. 3(a) given sparse data. Both Fig. 2(b) and Fig. 3(a) also suggest that the hidden data likelihood terms in CL-AIC and ICL played an insignificant role when the size of the data set is small. This is reflected by the fact that CL-AIC and AIC behaved very similarly given sparse data, so did ICL and BIC.

When the data set is densely populated, the three Gaussians are all well supported by the data samples. Therefore all scoring functions have a much easier task to estimate the number of hidden states. However, with the randomly distributed noise, the distribution of the data contained in each original Gaussian can become non-Gaussian. We know that a non-Gaussian distribution can be approximated using multiple Gaussians, which will then result in more than three hidden states in the estimated model structure. This explains why the models selected by VB, BIC, and AIC are slightly overfitted. Nevertheless, Fig. 2 shows that CL-AIC and ICL do not suffer from the same problem. This is because for both of them, complete likelihood is part of the scoring function, which means that an estimated model must explain explicitly the hidden state sequence as well. The temporal order of the hidden states that correspond to the noise-caused extra Gaussians cannot be well explained by the data. For instance, if two Gaussians are now used to approximate an original single Gaussian, the temporal order of these two Gaussians will be random. This results in low likelihood of the hidden data. In other words, a model that is more complicated than the true one would be penalised by both the penalty term and the term corresponding to the likelihood of the hidden data in CL-AIC and ICL. Therefore, both ICL and CL-AIC give more accurate estimates when sample size is large given noisy data compared to the other three scoring functions.

Both Fig. 2 and Fig. 3 show that VB is sensitive to initialisation even with densely sampled data (see the error bar in Fig. 2(a) and the local minima on the plots corresponding to VB in Fig. 3). Note that the key difference between computing VB and the other four scoring functions is that the former is based a modified EM algorithm and the latter use standard EM algorithm [5]. The modified EM algorithm involves computing a distribution over the model parameters rather than a point estimate of them in the M-step. This makes VB even more sensitive to initialisation than the other four. Consequently, both the plots in Fig. 3(a)

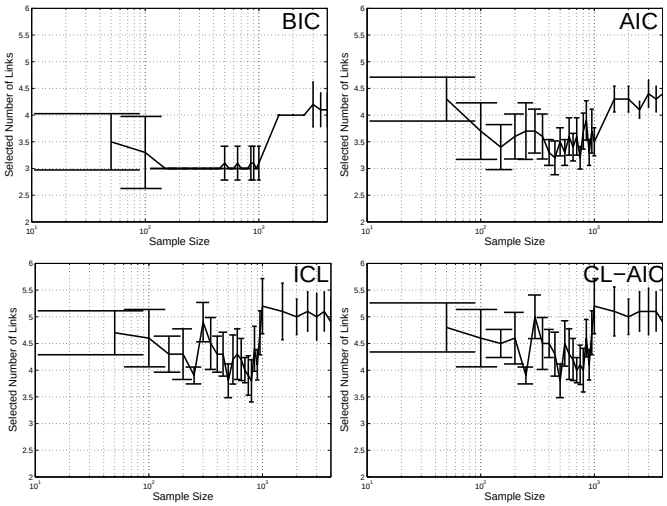
and (b) that correspond to VB are less smooth with more local minima than the others.

2.2 Determining the Topology of a DML-HMM

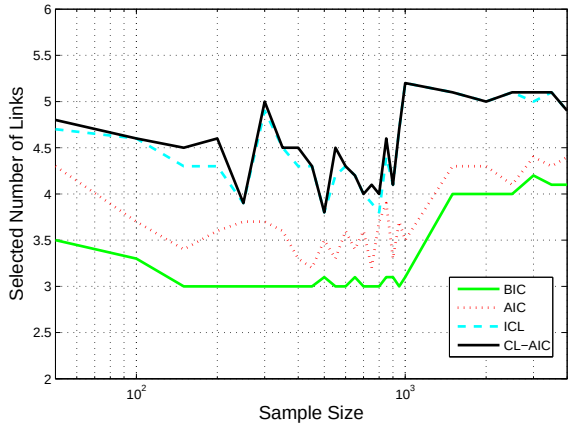
We now consider the problem of determining the unknown topology of a Dynamically Multi-Linked HMM (DML-HMM) [19] from data using CL-AIC as the scoring function. DML-HMMs were first proposed in [19] to overcome a limitation of HMMs in modelling complex dynamic processes. In particular, a HMM requires a large number of parameters to describe if it is to model multiple temporal processes simultaneously. This implies that a single hidden state variable and a single observation variable are to represent implicitly multiple sources of variations at any given time instance. Unless the training data set is very large and relatively 'clean', poor model learning is expected. To address this problem, various topological extensions to the standard HMMs can be considered to factorise explicitly the observation and/or state space by introducing multiple hidden state variables and multiple observation variables for modelling different temporal processes explicitly and simultaneously. Instead of being fully connected as in the case of a Coupled HMM (CHMM) [13] or being without any inter-links between different temporal processes as in the case of a Parallel HMM (PaHMM) [32], a DML-HMM aims to *only* connect a subset of relevant hidden state variables across multiple temporal processes. This is in order to learn the optimal factorisation of the state space and more importantly to be less sensitive to data noise. Instead of being fixed as in the cases of CHMMs and PaHMMs, the structure of a DML-HMM is learned from data. Given a data set, it is assumed that at each time instance the temporal process responsible for each data sample is known and the number of hidden states for each hidden variable is also known. The unknown structure to be learned is thus the topology of the graph, i.e. the links among different hidden nodes within two consecutive time instances.

To learn the optimal topology of a DML-HMM, CL-AIC is computed for each candidate topology using Eqn. (6) where the subscript K becomes the index of different topologies. The first and second terms on the right hand side of Eqn. (6) can be computed using an extended forward-backward algorithm for a DML-HMM. A detailed description of the algorithm can be found in the appendix of this paper. Note that the total number of candidate topologies K_{max} is exponential in the number of temporal processes N_t. Each candidate topology can be represented using a $N_t \times N_t$ inter-connection matrix whose elements have value 1 if there is a directed link between the two corresponding hidden nodes within two consecutive time instances and 0 otherwise.

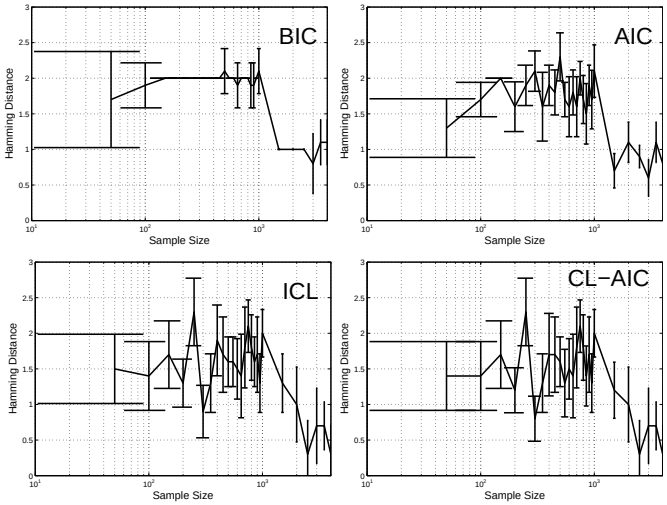
Synthetic experiments were carried out to examine the effectiveness of CL-AIC in comparison with



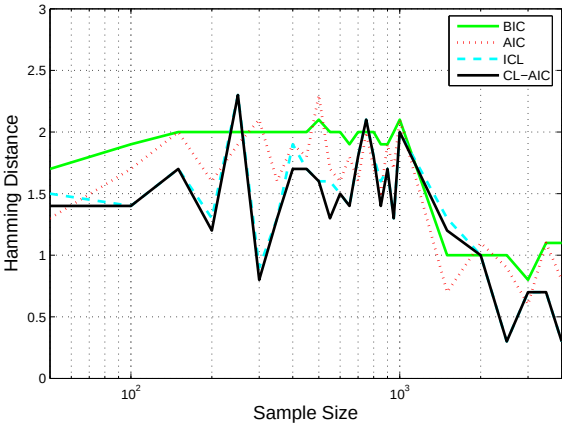
(a) Selected number of links (mean and ± 1 standard deviation over 50 trials)



(b) Mean of selected number of links (true number is 5)



(c) Hamming distance (mean and ± 1 standard deviation over 50 trials)



(d) Mean of hamming distance

Figure 4: Synthetic data experiment results for determining the topology of a DML-HMM using different scoring functions. In (a) & (b), the true number of links is 5. In (c) & (d), smaller Hamming distances indicate more accurate structure learning.

that of Bayesian Information Criterion (BIC), Akaike's Information Criterion (AIC), Integrated Completed Likelihood (ICL) on determining the topology of a DML-HMM given data of different sample sizes [3]. Data sets were randomly generated using a DML-HMM with three temporal processes whose topology is shown in Fig. 1(c). The model inter-connection matrix is thus $\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$. Both the observation and hidden variables are discrete with three possible values. The total number of free parameters of the model is therefore 66. The model parameters were set randomly. One fourth of the observational data were replaced by random numbers to synthesise noise contained in the observation. DML-HMMs with $K_{max} = 64$

[3] Variational Bayesian (VB) was not evaluated here because implementing VB for a dynamic graphical model with multiple temporal processes such as a DML-HMM is not trivial and beyond the scope of this paper.

different topologies were evaluated by four different scoring functions using data sets with sample size T varying from 25 to 4000. The performance of different scoring functions was measured by looking at both the number of links connecting hidden nodes within two consecutive time instances (the true number is 5) and the hamming distance between the estimated inter-connection matrices and the true one (the distance is zero if the structure is estimated correctly). The former measures model complexity while the latter measures the accuracy of the learned structure. The experimental results were obtained over 50 trials, and are shown in Fig. 4.

It can be seen from Fig. 4 that given sparse and noisy data the optimal models selected using all four different scoring functions tended to underfit with ICL and CL-AIC outperforming the other two. As the sample sizes increased, the optimal number of links among hidden nodes selected by CL-AIC and ICL converged towards the true number 5, while those selected by BIC and AIC converged to 4, (i.e. underfitting). In the meantime, the hamming distance obtained using different scoring functions decreased, with that obtained using CL-AIC being the smallest.

Our experiment shows that all four scoring functions are sensitive to initialisation of the model parameters (see the large standard deviations depicted in Fig. 4 (a) & (c)). The main reason for this shortcoming is that the EM algorithm used for learning model parameters and estimating the likelihood terms in different scoring functions is sensitive to initialisation. This is well reported in the literature. Although we have taken measures to avoid this problem by randomly initialising the EM algorithm and choose a solution that yields the highest likelihood, the problem is not totally eliminated. This suggests that each scoring function has a tendency to form a complicated surface with a large number of local minima with respect to the candidate model topologies. To solve this problem, more sophisticated method needs be exploited/developed.

Apart from this common limitation, the results also show that BIC and AIC tend to underfit the model whist CL-AIC and ICL do not suffer from that problem given sufficient data. There is an intuitive explanation for this. Essentially for both BIC and AIC, the likelihood of the observed data would not benefit directly from having interlinks between different time processes. In the meantime, the penalty terms in their formulation would increase exponentially with the increase of the number of interlinks. It is therefore natural to observe that underfitted models are preferred by BIC and AIC even given sufficient data. As for CL-AIC and ICL, they do not have the same problem because the likelihood of the inferred hidden state sequence is included in their formulation. Having multiple interlinks, especially those correct ones, helps greatly in explaining away the temporal occurrence of different states and therefore leads to higher likelihood of the hidden data.

3 Surveillance Video Segmentation

HMMs have been widely used for automatic segmentation of sequential/time-series data such as speech [14], DNA sequences [10], and video [8, 12, 33]. Here we propose to use HMM for content based surveillance video segmentation, i.e. to segment a continuous surveillance video based on activities captured in the video. Note that since there is only one video shot in a continuous surveillance video, the conventional shot-change detection based segmentation approach [2] cannot be adopted. We thus employed a video content trajectory based method. Specifically, a L-dimensional feature vector is first extracted from each image frame. The video content is thus represented a video trajectory in this L-dimensional feature space. This feature vector is then represented as the observational input of a HMM at each time instance. The conditional probability distributions (CPDs) of each observation variable are Gaussian for each of the K states of its parent hidden variable. The video content is then monitored using the discrete hidden variables in the model. The changes of video content can thus be detected as the changes of hidden states which correspond to breakpoints on a video trajectory (N detected change points/breakpoints result in N+1 video segments for a continuous video). Using a left-to-right HMM model [25], the number of hidden states would correspond to the number of video segments which can be automatically determined using CL-AIC.

3.1 Activity-based Video Content Representation

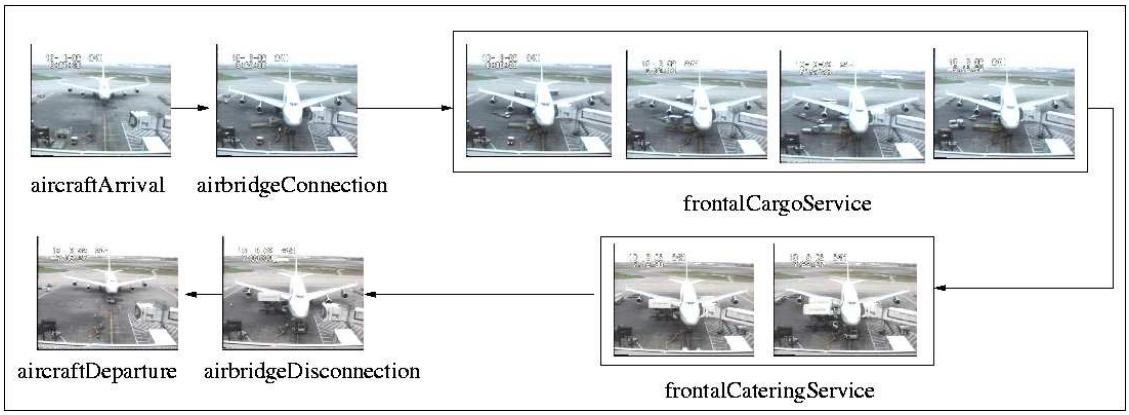


Figure 5: An example structure of a complete aircraft docking video. Representative frames of different activities occurred in the scene are also shown.

For clarity and concreteness, we shall illustrate our approach using surveillance videos monitoring aircraft docking operations at an airport. The total number of activities taking place in the aircraft docking area was decided as 15 by airport ground control officers. A trained ground control officer in the control room is required to monitor these 15 activities via both video inputs from multiple cameras and audio communi-

cations with various staff members on the ground. However, this does not imply that all of the 15 activities to be monitored by ground control officers are identifiable when CCTV video is the only input. It turns out that, when watching video for identifying these 15 activities on the list, most of them are completely or partially blocked by the aircraft and the air bridge (e.g crew bus picking up crew, engineers doing service check) and only 6 activities on the list are visually detectable. This is because the data were captured using a single camera from the frontal view. Obviously if more cameras were deployed to cover different angles, the number of detectable activities will increase. But it is still unlikely that all 15 activities will be visually detectable due to the limit of the cameras one can possibly install in reality and also the long distance from the camera to the objects of interest resulting in low image resolution.

Figure 5 shows an example activity based structure of a complete aircraft docking operation. One activity can be followed by either another activity immediately or a period of inactivity when no meaningful visual changes can be detected. The durations of activities and inactivity gaps vary hugely during a single aircraft docking operation and across different operations. There are also variations in the temporal order of activities and inactivity gaps. For example, the temporal order of `frontalCargoService` and `frontalCateringService` appears to be arbitrary. It is noted that some activities such as `airCraftArrival` involve the movement of a single object while other activities such as `frontalCargoService` and `frontalCateringService` consist of the movement of multiple objects which may leave and reappear in the scene. For the latter case, there often exist a number of short inactivity break-ups within an activity which are different from a long inactivity gap between two activities only by their durations. All these characteristics make analysis of surveillance video content very challenging.

We consider an activity based video content representation [36]. Visual events are detected and classified automatically in the scene. The semantics of video content are considered to be best encoded in the occurrence of such events and the temporal correlations among them. Firstly, an adaptive Gaussian mixture background model [31] is adopted to detect foreground pixels which are modelled using Pixel Change History (PCH) [36]. Secondly, the foreground pixels in vicinity are grouped into a blob using the connected component method. Each blob with its average PCH value greater than a threshold is then defined as an event. A detected event is represented as a 7-dimensional feature vector

$$\mathbf{v} = [\bar{x}, \bar{y}, w, h, R_f, M_p x, M_p y] \tag{9}$$

where $(\bar{x}, \bar{y})$ is the centroid of the blob, (w, h) is the blob dimension, R_f is the filling ratio of foreground

pixels within the bounding box associated with the blob, and $(M_p x, M_p y)$ are a pair of first order moments of the blob represented by PCH. Among these features, $(\bar{x}, \bar{y})$ are location features, (w, h) and R_f are principally shape features but also contain some indirect motion information, and $(M_p x, M_p y)$ are motion features capturing the direction of object motion. Classification is then performed in the 7D event feature space over a collection of training videos using a Gaussian Mixture Model (GMM). The number of event classes captured in videos is determined using CL-AIC (see [34] for the formulation of CL-AIC for GMMs). For the aircraft docking scene videos, 8 classes of events were automatically detected. It is worth mentioning that different events occur simultaneously and such an event detection mechanism makes mistakes. Mis-detection and wrong labelling can be caused by discontinuous movement and closeness of different objects. An example of event detection and classification is shown in Figure 6.

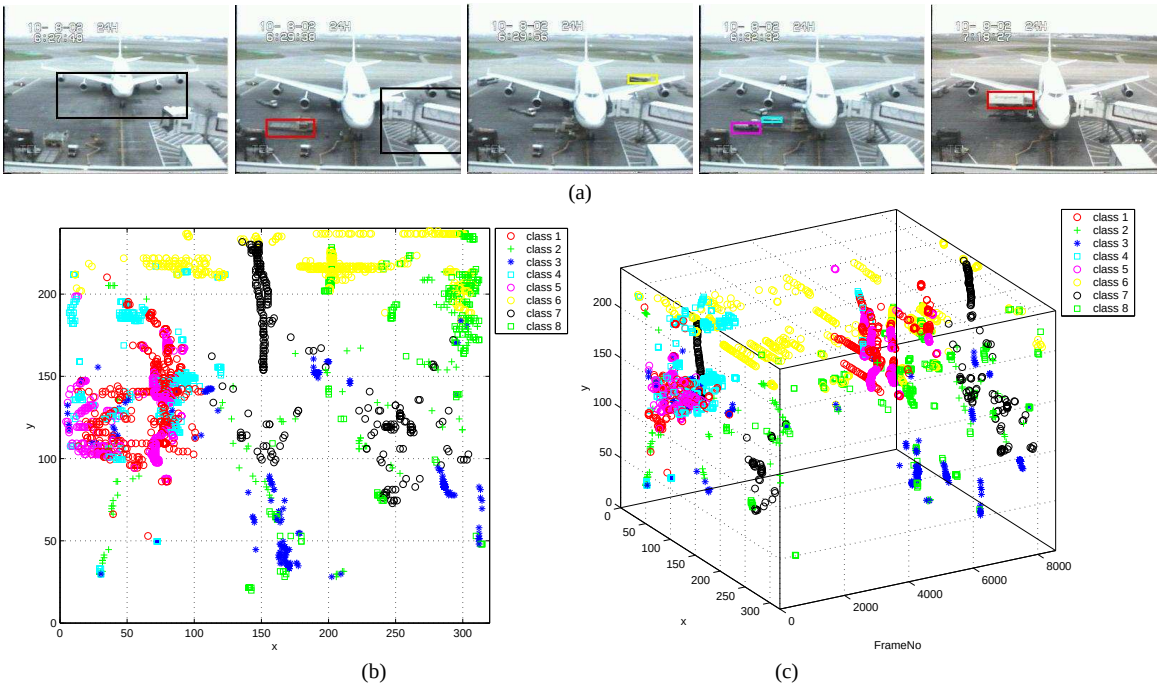


Figure 6: Event detection and classification in an aircraft docking scene video. Eight classes of events are detected automatically, each of which corresponds to different movement patterns in the image frame. For example, event class 7 corresponds to the movements of the aircraft. Event classes 4, 5, and 6 correspond to the movements of objects involved in activities `frontalCargoService` and `frontalCateringService`. (a) Events are detected and classified into different classes which are highlighted using bounding boxes in different colours. The spatial and temporal distribution of events of different classes are illustrated in (b) and (c) respectively, with centroids of different classes of events depicted using different colours.

Classified events can be considered as 'snapshots' of activities captured in a scene. To hide the image feature information and focus on the semantic video content, a scene vector is constructed for each image

frame of a video. A scene vector $\mathbf{sv}_t$ for a video frame t is defined as:

$$\mathbf{sv}_t = \left[s_t^1, ..., s_t^k, ..., s_t^K\right] \tag{10}$$

where K is the number of event classes automatically determined using CL-AIC. The value of s_t^k is the number of events of the k^{th} event class detected in the frame t. A scene vector gives a description of 'what is happening' in the scene through the class labels of the detected events. It is thus a concise representation of the video content at the semantic level (see Figure 7 for an example). However, directly using this scene vector to detect video content changes can cause problems for video segmentation. Specifically, the value of a scene vector $\mathbf{sv}_t$ can become $\mathbf{0}$ (i.e. absent of any event at a given frame) frequently throughout the video sequence (see Figure 7). This can be caused by either frequent but short inactivity break-ups within activities or long inactivities between activities. Each 'coming to zero' is reflected as a dramatic change in the value of $\mathbf{sv}_t$ due to the discrete nature of s_t^k. Those changes that correspond to real changes of video content can thus easily be overwhelmed by changes caused by the inactivity break-ups within activities, which makes temporal segmentation of the video difficult.

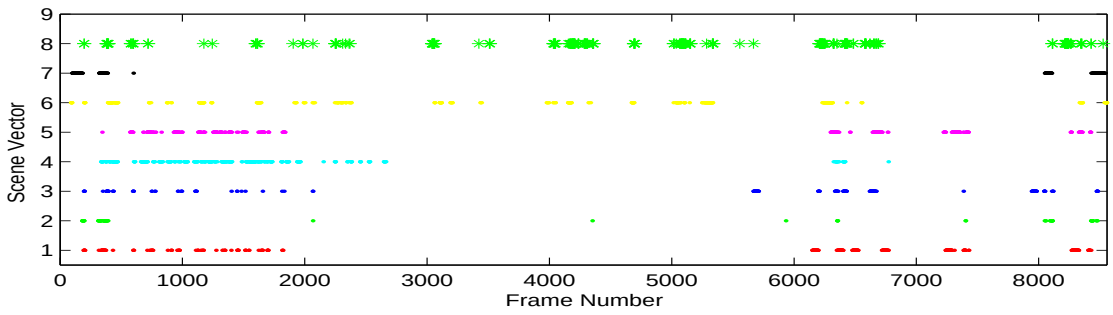


Figure 7: The example video shown in Figure 6 is represented by scene vectors evolving over time. We have $K = 8$ for this video. s_t^k is depicted by a dot in colour when $s_t^k > 0$. There are frequent but short inactivity break-ups within activities (see between frame 1 and frame 2000) and a long inactivity gaps between activities (between frame 3000 and frame 6000).

To overcome this problem, let us now consider representing the video content using a cumulative scene vector computed at frame t using $\mathbf{sv}_t$ from frame 1 to the frame t. More specifically, the k^{th} element of the cumulative scene vector (denoted as $\widetilde{\mathbf{sv}}_t$) is computed as:

$$\widetilde{s}_t^k = \sum_{i=1}^{t} s_i^k \tag{11}$$

The value of each element of $\widetilde{\mathbf{sv}}_t$ becomes continuous and will increase monotonically with time (see Figure 8). Compared to the scene vector representation $\mathbf{sv}_t$ (see Figure 7), the short inactivity break-ups at

individual frames have little impact on the values of the cumulative scene vector evolved over time. It thus becomes easier to detect breakpoints that correspond to significant changes in video content.

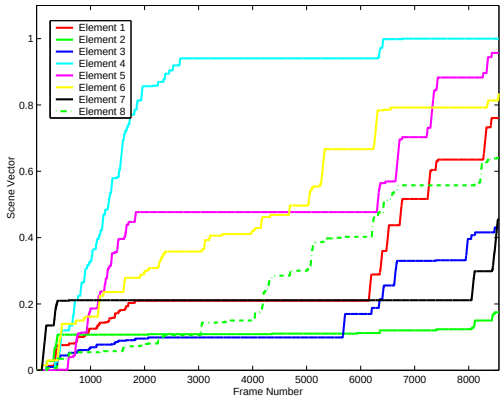


Figure 8: The 8 elements of $\widetilde{\mathbf{sv}}_t$ over time for the example video shown in Figure 6. Each element of $\widetilde{\mathbf{sv}}_t$ is normalised to have a value range of $[0, 1]$.

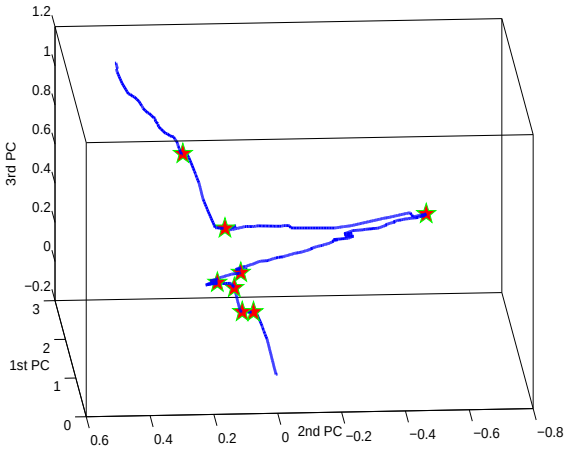
3.2 Determining the Number of Video Segments

Now each image frame of the aircraft docking scene videos is represented as a 8-dimensional feature vector which is then utilised as observational input for an HMM for temporal segmentation. The problem to be solved here is to automatically determine K, the number of hidden states which corresponds to the number of video segments.

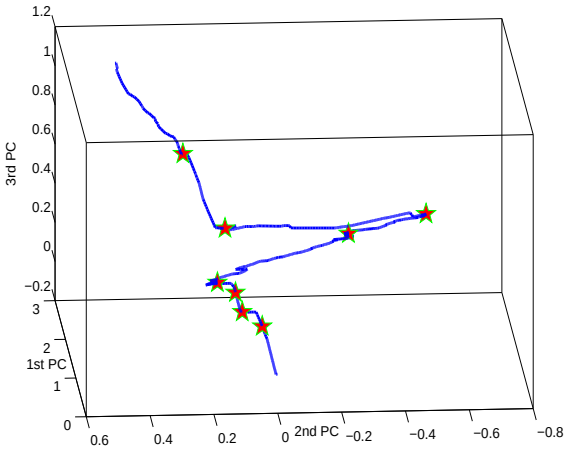
Our data set consists of 7 aircraft docking scene videos captured using a fixed CCTV analogue camera. After digitalisation, the final video sequences have a frame rate of 2Hz. Note that it is not uncommon to have such an extremely low frame rate for CCTV surveillance videos, which makes the video segmentation problem even more challenging. Each image frame has a size of 320×240 pixels. The 7 video sequences of aircraft docking lasted from 6470 to 17262 frames per sequence (around 50 to 140 minutes of recording), giving in total 72776 frames (10 hours) of video data that cover different times of different days under changing lighting conditions, from early morning, midday to late afternoon. They are referred as video 1 to video 7 respectively. The 7 videos were first manually segmented into activities to give the ground truth of the breakpoints for segmentation, resulting in a total of 64 breakpoints and 71 segments. The lengths of these video segments were within the range of 127 to 3210 frames.

The performance of different score functions are compared by looking at the number of detected breakpoints, the number of true positives, false positives, and false negatives. The results are shown in Table 1 and Fig. 9. Given the true number of breakpoints 64, it can be seen from Table 1 that both BIC, ICL, and VB

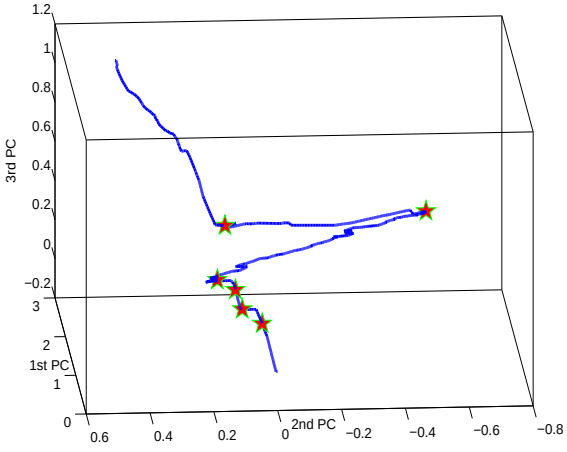
(a) Different activities captured in video 2



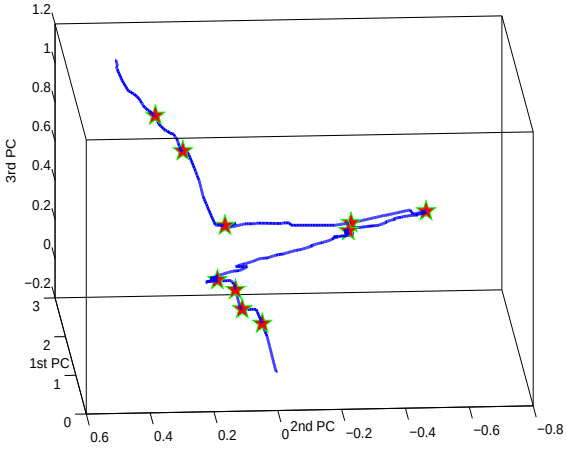
(b) Ground truth (8 breakpoints)



(c) CL-AIC (8 breakpoints)



(d) BIC & ICL & VB (6 breakpoints)



(e) AIC (10 breakpoints)

Figure 9: Determining the number of video segments for an aircraft docking video (video 2 of the 7 videos) using a HMM with different score functions. (a) Representative frames of different activities captured video 2. These activities were (from left to right): `aircraftArrival`, `airbridgeConnection`, `frontalCargoService`, `frontalCateringService`, `airbridgeDisconnection`, and `aircraftDeparture`. (b) Ground truth obtained by manually segmenting the video. (c)-(e) segmentation results using different scoring functions with the detected breakpoints shown on the video trajectory. Note that in (b)-(e) the video trajectories are shown in a 3D PCA space of the original 8D video content feature space just for the illustration purpose (the computation was done in the original 8D feature space).

	BIC	AIC	ICL	CL-AIC	VB
# Det. B. points	49	73	45	**62**	52
# True Positives	39	52	37	**54**	40
# False Positives	10	21	8	**8**	12
# False Negatives	15	12	17	**10**	14

Table 1: Comparing different scoring functions for video segmentation. The true number of breakpoints was 64.

underestimated the number of segments while AIC overestimated the segment number. In the meantime, the number of segments estimated using CL-AIC was the closest to the true number. On the accuracy of breakpoint detection, Table 1 shows that CL-AIC yielded the highest true positive number and lowest false positive and false negative numbers. In the meantime, both BIC, ICL, and VB gave low false positive number but low true positive number as well. As for AIC, high true positive number was obtained at the price of high false positive number.

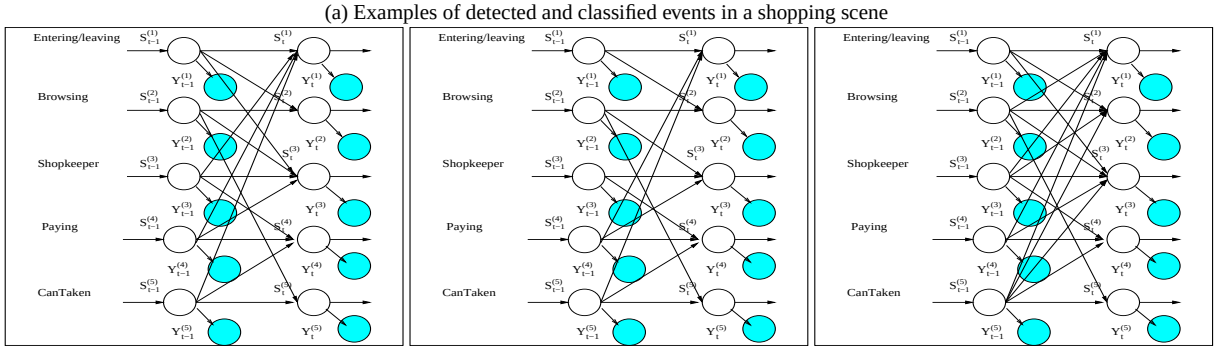
4 Discovering Causal Relationships in Group Activity Modelling

A group activity involves multiple objects co-existing and interacting in a shared common space. Examples of group activities include 'people playing football' and 'shoppers checking out at a supermarket'. Group Activity modelling is concerned with not only modelling actions executed by different objects in isolation, but also the interactions and causal/temporal relationships among these actions. Adopting a DML-HMM based activity modelling approach [19], we consider that a group activity is composed of different classes of dynamically linked visual events representing significant changes in the image over time caused by different objects in the scene. An event is represented by a multi-dimensional feature vector and automatically detected and classified into different event classes (see Section 3.1 for details). The detected events are then taken as the observational input to a DML-HMM so that learning causal and temporal relationships among different classes of events can be achieved by learning the optimal structure of the DML-HMM for modelling the dynamics of the detected events and the interactions among them. More specifically, each temporal process of the DML-HMM is used to model the dynamics of one class of events and those links among different processes capture the causal/temporal relationships of different classes of events. A inter-link between two temporal processes therefore indicate the existence of causality between the two corresponding classes of events.

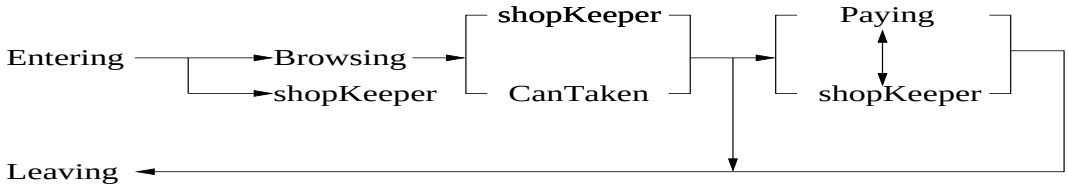
Experiments were carried out to compare the performance of different scoring functions for learning the structure of a DML-HMM and therefore the causal relationships among different classes of visual events. A simulated 'shopping scenario' was captured on a 20 minutes video. Some typical scenes can be seen in Fig. 10(a). The scene consists of a shopkeeper sitting behind a table on the right side of the view. Drink cans were laid out on a display table. Shoppers entered from the left and either browsed without paying or took a can and paid for it. The data used for this experiment were sampled at 5 frames per second with total number of 5699 frames of images sized 320×240 pixels.

In the 20 minutes video, a total of 4634 events were automatically detected and classified into 5 event classes. By observation, the 5 classes of events corresponds to 5 key constituents of the shopping activity. They were labelled as `canTaken`, `entering/leaving`, `shopkeeper`, `browsing` and `paying` respectively (see Fig. 10(a)). It was noted that different classes of events occurred simultaneously. It is also true that our event recognition model made errors. Some of the errors were caused by the occlusion, closeness and visual similarity among different events. Some others were due to the factor that the causal/temporal relationships among events were not considered at the level of event detection. For example, when a shopper stands in front of the shopkeeper, it is impossible to tell whether he/she is going to pay unless one takes into consideration whether any drink can was taken a moment ago. The event classifier is therefore expected to make such errors without taking into account the temporal and causal correlations among different classes of events. Such causal/temporal relationships are modelled using a DML-HMM.

(a) Examples of detected and classified events in a shopping scene



(b) CL-AIC (15 inter-links) (c) BIC & ICL (13 inter-links) (d) AIC (19 inter-links)



(e)

Figure 10: Discovering causal relationships among visual events in a shopping scene. In (a), events belonging to 5 event classes `canTaken`, `entering/leaving`, `shopkeeper`, `browsing` and `paying` are indicated with bounding boxes in magenta, red, blue, green and cyan respectively. (b)-(d): topologies of DML-HMMs learned using different scoring functions. (e): The expected causal and temporal structure of the shopping activity.

There are 5 temporal processes in this DML-HMM, each corresponding to one class of events. We also consider two states for each hidden state variable, i.e. a binary variable switching between the status of `True` and `False` which correspond to whether or not an event of a certain class is truly present in each

frame. Each observation variable is continuous and given by a 7D feature vector representing a detected event (see Eqn. (9)). Their distributions are mixtures of Gaussian with respect to the states of their discrete parent nodes. For model learning, the distributions of the detected events learned using GMMs are used to initialise the distributions of the observation vectors. The priors and transition matrices of states are initialised randomly. The number of candidate topologies for a 5-temporal-process DML-HMM is too large to be searched exhaustively. The Structural EM algorithm [17] was thus adopted to search for the optimal structure more efficiently using different scoring functions.

The discovered causal/temporal relationships among different classes of events are embodied in the learned topologies of the DML-HMMs. For instance, a link pointing from the `canTaken` process towards the `paying` process indicates the causality between these two classes of events. Compared with the expected structure of the shopping activity as shown in Fig. 10(e), it can be seen that the causal relationships among different classes of events and the temporal structure of the activity were discovered correctly by CL-AIC (Fig. 10(b)). In comparison, a over-complicated DML-HMM topology was selected using AIC (Fig. 10(d)) while both BIC and ICL underestimated the number of inter-links among different temporal processes (Fig. 10(c)), resulting in over-simplified causal relationships among different classes of events.

Note that in Section 3 and Section 4 the proposed scoring function is employed to address very different vision problems. In Section 3, the objective is to temporally segment a continuous surveillance video according to the activities taking place in the scene. As explained earlier, the number of activities that are visually identifiable and can be segmented from a video is determined by the nature of the camera view and the image resolution (i.e. the distance from the camera to the objects of interest). In Section 4, the objective is to discover the causal relationship among the detected events. Obviously if different number of events were detected (e.g. using different features), the set of causal relationships will be different as well. But what we are interested in is, for *a given* set of events (from whichever a representation has been chosen), whether the correct causal relationship can be recovered using different graphical model structure scoring functions. So the point here is not about what is the optimal number of events one should detect. It is about whether the causal relationship among the actually detected events can be learned correctly.

5 Discussion and Conclusion

Our experimental results show that the performance of CL-AIC on learning the structure of a dynamic graphical model with hidden variables is superior to that of existing popular alternatives including BIC,

AIC, ICL, and VB. This is especially true when the given data set is noisy and sparse. Similar results were reported in the case of static graphical models in [34]. However, it is interesting to note the difference in the definitions of 'data sparseness' in the context of DBNs and in that of static models such as GMMs. The sparseness of a data set is normally measured in comparison to the number of free parameters of a model. The experiments reported in [34] show that a sample size smaller than 5 times of the parameter number should be considered as sparse while our experiments on HMMs and DML-HMMs show that a sample size smaller than 20 times of the number of parameters would qualify for being sparse (see Fig. 2 and Fig. 4).

It is worth pointing out that our CL-AIC and the other four scoring functions evaluated in this paper can be classified into two categories based on the underlying principle of derivation. Both AIC and CL-AIC are derived based on information coding theory. Our experimental results suggest that the extra term in the formation of CL-AIC compared to AIC (the second term on the right hand side of Eqn. (4)) is able to rectify the overfitting tendency of AIC given densely distributed data. Different from AIC and CL-AIC, both BIC, ICL, and VB are derived as approximations to Bayesian Model Selection (BMS) [27] which aims to select a model that maximise the marginal likelihood for a data set given the model. Compared to BIC, the approximation made in the derivation of VB is less drastic therefore more accurate resulting in better structure learning. However, it also pays the price for losing generality - VB can only be applied to discrete graphical models whilst the other four can also be implemented for continuous ones such as PCA, ICA, Kalman Filters, and continuous HMMs.

It is also interesting to compare our CL-AIC with an recently proposed structure learning algorithm based on an entropic prior and parameter extinction [11]. The method has been applied to HMM for activity modelling [12]. However, despite being called a structure learning algorithm, it is essentially a parameter learning algorithm aiming to learn the optimal hidden state transition probabilities which are part of the model parameters rather than model structure. With the addition of state trimming, it is possible for the method to learn the optimal number of hidden states for a HMM. Nevertheless, as pointed out in [11], the algorithm tend to keep superfluous gating states in the learned structure. It therefore not suitable for determining the state number for a HMM. Moreover, it cannot be applied for more general dynamical graphical models such as DML-HMMs.

It is critical to distinguish the "expected model" (the model learned from data) and the true underlying model when studying a real-world model selection problem. In particular, one must note that an expected model and the real (unknown) model underpinning the original data are not necessary the same. We consider that one of the main causes of this discrepancy is that when solving a real-world data-modelling problem,

for computational tractability, one must first perform feature extraction and data pre-processing before a model can be built. However, if those features extracted are not a good representative of the original data, the estimated model, even if it corresponds well to the feature representation, is not a good model for explaining the original data. To make sure that a model does indeed capture accurately the underlying nature of the original data, the choice of feature selection and representation is crucial. We have done exactly that in the two real-data experiments presented in this paper. Specifically, the discrete event based video trajectory representation for the video content analysis has been shown to be more informative and robust to noise compared to conventional representations in our previous work [33]. We have also demonstrated that the event based activity representation is superior to other commonly used representations in revealing the causal/temporal relationships in [19]. We therefore argue with strong justification that for the experiments carried out in this work, the "expected" model does go some way to explain the true underlying nature of the original problem rather than merely a good explanation of the wanted results.

In conclusion, we have proposed a novel scoring function, CL-AIC for selecting the optimal structure of DBNs. The effectiveness of CL-AIC has been demonstrated on solving the challenging problem of video content analysis. In particular, it is evident from our results that CL-AIC is superior to the commonly used scoring functions including BIC, AIC, ICL, and VB, especially give sparse and noisy visual data. It is worth pointing out that either model parameter estimation or computing the posterior probabilities will become computational intractable for learning the structure of a large DBN. To address this problem, approximation methods such as stochastic simulation [22] or variational approximations [18] need to be adopted. Our ongoing work includes further investigation on how the performance of different scoring functions will be affected by the use of approximation methods.

Appendix

Computing CL-AIC for a DML-HMM

CL-AIC for a DML-HMM is formulated as the following:

$$\text{CL–AIC}(K) = -\log\left(\sum_{S} p(\mathcal{Y}|S,\hat{\boldsymbol{\theta}}_K)p(S|\hat{\boldsymbol{\theta}}_K)\right) - \log p(S=\hat{\mathcal{Z}}|\mathcal{Y},\hat{\boldsymbol{\theta}}_K) + C_K. \tag{12}$$

To compute the first and second terms on the right hand side of the above equation, the ML (maximum likelihood) estimation of the model parameters, $\hat{\boldsymbol{\theta}}_K$ has to be obtained first. This is achieved through an

extended dynamic programming algorithm similar to the forward-backward algorithm for a standard HMM.

Let us consider a DML-HMM with C temporal processes and one hidden variable and one observation variable respectively for each temporal process at each time instance. We thus have $N_h = N_o = C$ where N_h and N_o are the number of hidden state variables and observation variables at each time instance respectively. It is assumed that all the hidden state variables are discrete and all the observation variables are continuous whose probability density functions are Gaussian with respect to each state of their parent hidden state variables. The parameter space thus consists of the following components:

1. The initial state distribution $\pi = \{\pi_{i^{(c)}}\}$ where $\pi_{i^{(c)}} = P\left(S_1^{(c)} = q_{i^{(c)}}\right)$, $1 \leq i^{(c)} \leq N^{(c)}$ and $1 \leq c \leq C$.

2. The state transition probability distribution $\mathbf{A} = \left\{a_{Pa\left(j^{(c)}\right)j^{(c)}}\right\}$ where $a_{Pa\left(j^{(c)}\right)j^{(c)}} = P\left(S_{t+1}^{(c)} = q_{j^{(c)}} | Pa\left(S_{t+1}^{(c)}\right) = q_{Pa\left(j^{(c)}\right)}\right)$, $Pa\left(S_{t+1}^{(c)}\right)$ are the hidden variables at time t on which $S_{t+1}^{(c)}$ is conditionally dependent, $Pa\left(j^{(c)}\right)$ are subscripts of those discrete values that $Pa\left(S_{t+1}^{(c)}\right)$ can assume, $1 \leq j^{(c)} \leq N^{(c)}$ and $1 \leq c \leq C$.

3. The observation probability distribution $\mathbf{B} = \left\{b_{i^{(c)}}\left(\mathcal{Y}_t^{(c)}\right)\right\}$ where $b_{i^{(c)}}\left(\mathcal{Y}_t^{(c)}\right) = \mathcal{N}\left(\mathcal{Y}_t^{(c)}; \mu_{i^{(c)}}, \mathbf{U}_{i^{(c)}}\right)$, $\mu_{i^{(c)}}$ and $\mathbf{U}_{i^{(c)}}$ are the mean vector and covaria nce matrix of the normal (Gaussian) distribution with respect to $S_t^{(c)} = q_{i^{(c)}}$, $1 \leq i^{(c)} \leq N^{(c)}$ and $1 \leq c \leq C$.

Given an observation sequence $\mathcal{Y}$ and a model structure $\boldsymbol{\lambda}$, we need to determine the model parameters $\boldsymbol{\theta}(\boldsymbol{\lambda}) = \{\mathbf{A}, \mathbf{B}, \pi\}$ that maximise the probability of the observation sequence given the model structure $P(\mathcal{Y}|\boldsymbol{\lambda}, \boldsymbol{\theta}(\boldsymbol{\lambda}))$. There is no analytical solution to determine the optimal parameters given a finite observation sequence. However, the parameters can be estimated iteratively using dynamic programming. Let us first define the following variables:

- The forward variable $\alpha_t\left(i^{(1)}, ..., i^{(C)}\right) = P\left(\mathcal{Y}_1, \mathcal{Y}_2, ..., \mathcal{Y}_t, S_t^{(1)} = q_{i^{(1)}}, ..., S_t^{(C)} = q_{i^{(C)}} | \boldsymbol{\lambda}, \boldsymbol{\theta}(\boldsymbol{\lambda})\right)$, i.e., the probability of the partial observation sequence until time t and states for $S_t^{(1)}, ..., S_t^{(C)}$, given the model $\boldsymbol{\lambda}$ and $\boldsymbol{\theta}(\boldsymbol{\lambda})$:

$$\alpha_t\left(i^{(1)}, ..., i^{(C)}\right) = \begin{cases} \prod_{c=1}^{C} \pi_{i^{(c)}} b_{i^{(c)}}\left(\mathcal{Y}_t^{(c)}\right) & \text{if } t = 1 \\ \prod_{c=1}^{C} \left(\left(\sum_{j^{(1)},...,j^{(C)}} \alpha_{t-1}\left(j^{(1)}, ..., j^{(C)}\right) a_{Pa\left(i^{(c)}\right)i^{(c)}}\right) b_{i^{(c)}}\left(\mathcal{Y}_t^{(c)}\right)\right) & \text{if } 1 < t \leq T \end{cases}$$

- The backward variable

$\beta_t\left(i^{(1)},...,i^{(C)}\right) = P\left(\mathcal{Y}_t,...,\mathcal{Y}_T, S_t^{(1)} = q_{i^{(1)}},...,S_t^{(C)} = q_{i^{(C)}}|\boldsymbol{\lambda},\boldsymbol{\theta}(\boldsymbol{\lambda})\right)$, i.e., the probability of the partial observation sequence from $t+1$ to T, given the states for $S_t^{(1)},...,S_t^{(C)}$ and the model $\boldsymbol{\lambda}$ and $\boldsymbol{\theta}(\boldsymbol{\lambda})$:

$$\beta_t\left(i^{(1)},...,i^{(C)}\right) = \begin{cases} 1 & \text{if } t = T \\ \sum_{j^{(1)},...,j^{(C)}} \left(\prod_{c=1}^{C}\left(a_{Pa\left(j^{(c)}\right)j^{(c)}} b_{j^{(c)}}\left(\mathcal{Y}_t^{(c)}\right)\right)\beta_{t+1}\left(j^{(1)},...,j^{(C)}\right)\right) & \text{if } 1 \leq t < T \end{cases}$$

- $\xi_t\left(i^{(1)},...,i^{(C)},j^{(1)},...,j^{(C)}\right) = P\left(S_t^{(1)} = q_{i^{(1)}},...,S_t^{(C)} = q_{i^{(C)}},S_{t+1}^{(1)} = q_{j^{(1)}},...,S_{t+1}^{(C)} = q_{j^{(C)}}|\boldsymbol{\lambda},\boldsymbol{\theta}(\boldsymbol{\lambda})\right)$, i.e., the probability of being at certain states at time t and $t+1$, given the model and observation sequence:

$$\xi_t\left(i^{(1)},...,i^{(C)},j^{(1)},...,j^{(C)}\right) = \frac{\beta_{t+1}\left(j^{(1)},...,j^{(C)}\right)\prod_{c=1}^{C}\alpha_t\left(i^{(1)},...,i^{(C)}\right)a_{Pa\left(j^{(c)}\right)j^{(c)}}b_{j^{(c)}}\left(\mathcal{Y}_{t+1}^{(c)}\right)}{P(\mathcal{Y}|\boldsymbol{\lambda},\boldsymbol{\theta}(\boldsymbol{\lambda}))}$$

where $P(\mathcal{Y}|\boldsymbol{\lambda},\boldsymbol{\theta}(\boldsymbol{\lambda}))$ can be computed using the forward and backward variables:

$$P(\mathcal{Y}|\boldsymbol{\lambda},\boldsymbol{\theta}(\boldsymbol{\lambda})) = \sum_{i^{(1)},...,i^{(C)}} \alpha_t\left(i^{(1)},...,i^{(C)}\right)\beta_t\left(i^{(1)},...,i^{(C)}\right) \tag{13}$$

- $\gamma_t\left(i^{(1)},...,i^{(C)}\right) = P\left(S_t^{(1)} = q_{i^{(1)}},...,S_t^{(C)} = q_{i^{(C)}}|\mathcal{Y},\boldsymbol{\lambda},\boldsymbol{\theta}(\boldsymbol{\lambda})\right)$, i.e., the probability of being at certain states at time t, given the model and observation sequence:

$$\gamma_t\left(i^{(1)},...,i^{(C)}\right) = \frac{\alpha_t\left(i^{(1)},...,i^{(C)}\right)\beta_t\left(i^{(1)},...,i^{(C)}\right)}{\sum_{i^{(1)},...,i^{(C)}}\alpha_t\left(i^{(1)},...,i^{(C)}\right)\beta_t\left(i^{(1)},...,i^{(C)}\right)} \tag{14}$$

Denote the current parameter estimates as $\boldsymbol{\theta}(\boldsymbol{\lambda}) = \{\mathbf{A},\mathbf{B},\pi\}$, the re-estimated parameters $\bar{\boldsymbol{\theta}}(\boldsymbol{\lambda}) = \{\bar{\mathbf{A}},\bar{\mathbf{B}},\bar{\pi}\}$ can be computed using the following re-estimation formula: Denote the current parameter estimates as $\boldsymbol{\theta}(\boldsymbol{\lambda}) = \{\mathbf{A},\mathbf{B},\pi\}$, the re-estimated parameters $\bar{\boldsymbol{\theta}}(\boldsymbol{\lambda}) = \{\bar{\mathbf{A}},\bar{\mathbf{B}},\bar{\pi}\}$ can be computed using the following re-estimation formula:

$$\bar{\pi}_{i^{(c)}} = \sum_{i^{(1)},...,i^{(c-1)},i^{(c+1)},...,i^{(C)}} \gamma_1\left(i^{(1)},...,i^{(C)}\right) \tag{15}$$

$$\bar{a}_{Pa\left(j^{(c)}\right)j^{(c)}} = \frac{\sum_{t=1}^{T-1}\sum_{j^{(1)},...,j^{(c-1)},j^{(c+1)},...,j^{(C)},i^{(c')}\neq Pa\left(j^{(c)}\right)}\xi_t\left(i^{(1)},...,i^{(C)},j^{(1)},...,j^{(C)}\right)}{\sum_{t=1}^{T}\sum_{i^{(c')}\neq Pa\left(j^{(c)}\right)}\gamma_t\left(i^{(1)},...,i^{(C)}\right)} \tag{16}$$

$$\bar{\mu}_{i^{(c)}} = \frac{\sum_{t=1}^{T}\left(\sum_{i^{(1)},...,i^{(c-1)},i^{(c+1)},...,i^{(C)}}\gamma_t\left(i^{(1)},...,i^{(C)}\right)\right)\mathcal{Y}_t^{(c)}}{\sum_{t=1}^{T-1}\sum_{i^{(1)},...,i^{(c-1)},i^{(c+1)},...,i^{(C)}}\gamma_t\left(i^{(1)},...,i^{(C)}\right)} \tag{17}$$

$$\bar{\mathbf{U}}_{i^{(c)}} = \frac{\sum_{t=1}^{T}\left(\sum_{i^{(1)},...,i^{(c-1)},i^{(c+1)},...,i^{(C)}}\gamma_t\left(i^{(1)},...,i^{(C)}\right)\right)\left(\mathcal{Y}_t^{(c)}-\mu_{i^{(c)}}\right)\left(\mathcal{Y}_t^{(c)}-\mu_{i^{(c)}}\right)^T}{\sum_{t=1}^{T}\sum_{i^{(1)},...,i^{(c-1)},i^{(c+1)},...,i^{(C)}}\gamma_t\left(i^{(1)},...,i^{(C)}\right)} \tag{18}$$

If we iteratively use $\bar{\boldsymbol{\theta}}(\boldsymbol{\lambda})$ to replace $\boldsymbol{\theta}(\boldsymbol{\lambda})$ and repeat the re-estimation calculation until some limiting point is reached, the final result is a maximum likelihood estimate of parameters $\hat{\boldsymbol{\theta}}(\boldsymbol{\lambda})$.

Now given the learned parameters $\hat{\boldsymbol{\theta}}(\boldsymbol{\lambda})$, the first term on the right hand side of Eqn. (12) can be computed using Eqn. (13). The remaining problem is to compute the second term which involves the estimation of the most probable hidden state sequence. We formulate an extended Viterbi algorithm to infer the sequence given the observation and learned parameters. Let us first define the following variables:

- $\delta_t\left(i^{(1)},...,i^{(C)}\right) = \max_{\{S_1^c\},...,\{S_{t-1}^c\}} P\left(\{S_1^{(c)}\},...,\{S_{t-1}^{(c)}\},S_t^{(1)}=q_{i^{(1)}},...,S_t^{(C)}=q_{i^{(C)}},\mathcal{Y}_1,...,\mathcal{Y}_t|\boldsymbol{\lambda},\boldsymbol{\theta}(\boldsymbol{\lambda})\right)$ where $\{S_t^c\}=\{S_t^{(1)},...,S_t^{(C)}\}$. $\delta_t\left(i^{(1)},...,i^{(C)}\right)$ is the highest probability of the partial observation sequence until time t and a sequences of hidden states from time 1 to time t, given the model $\boldsymbol{\lambda}$ and $\boldsymbol{\theta}(\boldsymbol{\lambda})$.
- $\varphi_t\left(i^{(1)},...,i^{(C)}\right)$, which is an array used to store the best state sequence. The best hidden states at time t for the C hidden state variables are denoted as $\{S_t^{*(c)}\}$.

The extended Viterbi algorithm has the following steps:

1. Initialisation:

$$\begin{aligned}\delta_1\left(i^{(1)},...,i^{(C)}\right) &= \prod_{c=1}^{C}\pi_{i^{(c)}}b_{i^{(c)}}\left(\mathcal{Y}_t^{(c)}\right)\\ \varphi_1\left(i^{(1)},...,i^{(C)}\right) &= \mathbf{0}\end{aligned}$$

2. Recursion:

$$\begin{aligned}
\delta_t\left(i^{(1)},...,i^{(C)}\right) &= \max_{j^{(1)},...,j^{(C)}}\left\{\prod_{c=1}^{C}\delta_{t-1}\left(j^{(1)},...,j^{(C)}\right)a_{Pa\left(i^{(c)}\right)i^{(c)}}b_{i^{(c)}}\left(\mathcal{Y}_t^{(c)}\right)\right\} \\
\varphi_t\left(i^{(1)},...,i^{(C)}\right) &= \arg\max_{j^{(1)},...,j^{(C)}}\left\{\prod_{c=1}^{C}\delta_{t-1}\left(j^{(1)},...,j^{(C)}\right)a_{Pa\left(i^{(c)}\right)i^{(c)}}b_{i^{(c)}}\left(\mathcal{Y}_t^{(c)}\right)\right\}
\end{aligned}$$

where $1 < t \leq T$.

3. Termination:

$$\{S_T^{*(c)}\} = \arg\max_{i^{(1)},...,i^{(C)}}\delta_t\left(i^{(1)},...,i^{(C)}\right)$$

4. Best state sequence backtracking:

$$\{S_t^{*(c)}\} = \varphi_{t+1}\left(\{S_{t+1}^{*(c)}\}\right)$$

where $t = T-1, T-2, ..., 1$.

Given the most probable hidden states sequence $\hat{\mathcal{Z}} = \{S_t^{*(c)}\}$, the second term of Eqn. (12) is computed as:

$$\log p(S = \hat{\mathcal{Z}}|\mathcal{Y}, \hat{\boldsymbol{\theta}}_K) = \sum_{t=1}^{T}\log\gamma_t \tag{19}$$

where γ_t, computed using Eqn. (14), is the probability of being at the most probable hidden states sequence at time t given the model and observation sequence.

References

[1] H. Akaike. Information theory and an extension of the maximum likelihood principle. In *2nd International Symposium on Information Theory*, pages 267–281, 1973.

[2] N. Babaguchi, Y. Kawai, and T. Kitahashi. Event based indexing of broadcasting sports video by intermodal collaboration. *IEEE transactions on Multimedia*, 4(1):68–75, 2002.

[3] L.E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state markov chains. *Ann. Math. Stat.*, 37:1554–1563, 1966.

[4] M. Beal and Z. Ghahramani. The variational bayesian em algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian Statistics*, 7, 2003.

[5] M. J. Beal and Z. Ghahramani. Variational bayesian learning of directed graphical models with hidden variables. *Bayesian Analysis*, 1:1–44, 2006.

[6] C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):719–725, 2000.

[7] C. Biernacki and G. Govaert. Using the classification likelihood to choose the number of clusters,. *Computing Science and Statistics*, 29:451–457, 1997.

[8] J. Boreczky and L. Wilcox. A hidden markov model framework for video segmentation using audio and image features. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 3741–3744, 1998.

[9] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in bayesian networks. In *Uncertainty in AI*, pages 115–123, 1996.

[10] R. Boys. and D. Henderson. A bayesian approach to dna sequence segmentation. *Biometrics*, 60:573–588, 2004.

[11] M. Brand. Structure discovery in conditional probability models via an entropic prior and parameter extinction. *Neural Computation*, 11(5):1155–1182, 1999.

[12] M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):844–851, 2000.

[13] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 994–999, Puerto Rico, 1996.

[14] F. Brugnara, D. Falavigna, and M. Omologo. Automatic segmentation and labeling of speech based on hidden markov models. *Speech Communication*, 12:357–370, 1993.

[15] T. Duong, H. Bui, D. Phung, and S. Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 838–845, 2005.

[16] G.D. Forney. The Viterbi algorithm. *Proceedings of the IEEE*, 61:268–278, 1973.

[17] N. Friedman, K. Murphy, and S. Russell. Learning the structure of dynamic probabilistic networks. In *Uncertainty in AI*, pages 139–147, 1998.

[18] Z. Ghahramani. Learning dynamic bayesian networks. In *Adaptive Processing of Sequences and Data Structures. Lecture Notes in AI*, pages 168–197, 1998.

[19] S. Gong and T. Xiang. Recognition of group activities using dynamic probabilistic networks. In *IEEE International Conference on Computer Vision*, pages 742–749, 2003.

[20] D. Heckerman, D. Geiger, and D. Chickering. Learning bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

[21] N. Johnson, A. Galata, and D. Hogg. The acquisition and use of interaction behaviour models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 866–871, 1998.

[22] K. Kanazawa, D. Koller, and S. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In *Uncertainty in AI*, pages 346–351, 1995.

[23] R. Kass and A. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90:377–395, 1995.

[24] S. Kullback. *Information theory and statistics*. Dover: New York, 1968.

[25] L.R.Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[26] N. Oliver, B. Rosario, and A. Pentland. A bayesian computer vision system for modelling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, August 2000.

[27] A. Raftery. Bayes model selection in social research. *Sociological Methodology*, 90:181–196, 1995.

[28] J. Rissanen. *Stochastic Complexity in Statistical Inquiry*. World Scentific, 1989.

[29] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

[30] R. Shibata. Selection of the order of an autoregressive model by Akaike's Information Criterion. *Biometrika*, 63:117–126, 1976.

[31] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–758, August 2000.

[32] C. Vogler and D. Metaxas. A framework for recognizing the simultaneous aspects of american sign language. *Computer Vision and Image Understanding*, 81:358–384, 2001.

[33] T. Xiang and S. Gong. Activity based video content trajectory representation and segmentation. In *British Machine Vision Conference*, pages 177–186, 2004.

[34] T. Xiang and S. Gong. Visual learning given sparse data of unknown complexity. In *IEEE International Conference on Computer Vision*, pages 701–708, 2005.

[35] T. Xiang and S. Gong. Optimal dynamic graphs for video content analysis. In *British Machine Vision Conference*, 2006.

[36] T. Xiang, S. Gong, and D. Parkinson. Autonomous visual events detection and classification without explicit object-centred segmentation and tracking. In *British Machine Vision Conference*, pages 233–242, 2002.