Modeling Influence Between Experts

Wen Dong and Alex Pentland

E15-383, 20 Ames Street
The MIT Media Laboratory
Cambridge, MA, 02139-4307
{wdong, sandy}@media.mit.edu

Abstract. A common problem of ubiquitous sensor-network computing is combining evidence between multiple agents or experts. We demonstrate that the *latent structure influence model*, our novel formulation for combining evidence from multiple dynamic classification processes ("experts"), can achieve greater accuracy, efficiency, and robustness to data corruption than standard methods such as HMMs. It accomplishes this by simultaneously modeling the structure of interaction and the latent states.

1 Introduction

Human computing is the next generation human-computer interaction scheme [8]. In this scheme, a multitude of unobtrusive and ubiquitous sensors work with each other, intelligently sense human behavior and interaction, and provide assistance accordingly. The *human computing* paradigm is different from the traditional keyboard/mouse multimodal scheme: The traditional scheme is computer-centered, and requires considerable efforts from a human user in order to make a computer understand. In comparison, the human computing scheme is human based. It requires the intelligent sensors to predict human behavior and interaction precisely without even being noticeable by a human user, in order to provide the best assistance possible.

Combining evidence from different sensors, classifiers, agents, or experts is a major algorithmic challenge in human computing. We propose the influence model as a new, efficient, and robust method for combining evidence from different dynamic processes. The influence model is analogous to the work of a team of experts. In this team, different experts cope with different types of data and use different statistical models. Each expert consults with the other experts about their results, instead of their raw data, to form a better understanding of the situation. The experts find others whose results have information about their particular problem, and form networks that can be used to pool data, identify outliers, etc.

The influence model has proven to be an efficient, robust method for analyzing multi-expert dynamics problems. It is in the tradition of N-heads dynamic programming on coupled hidden Markov models [7], the observable structure influence model [1], and the partially observable influence model [2], but extends these previous models by providing greater generality, accuracy, and efficiency.

T.S. Huang et al. (Eds.): AI for Human Computing, LNAI 4451, pp. 170–189, 2007.

In this introductory section, we will first outline two types of several experimental platforms where we have used the influence modeling methodology. This will allow us to sketch some of the opportunities and challenges for our method of combining experts. We believe the same method can be applied to other human computing scenarios, and can be a good candidate for combining evidence of human physiological signals as well as social signals.

Example Applications

Our first example illustrating the general problem of combining expert classifiers is the *inSense* system [3] which combined several wearable sensor systems, each of which attempted to classify the state of the wearer. The ultimate goal of this system was to combine these local experts into a global estimate of wearer's state, and use this to control data collection by camera and microphone. The system was developed as part of the DARPA *Advanced Soldier Sensor Information System and Technology* (*ASSIST*) program [4].

In the *inSense* system (see Figure 1), data are collected by two accelerometers (worn on left hip and right wrist), an ambient audio recorder, and a camera (the latter two worn on the chest). From the data, four within-category "experts" are able to recognize various types of wearer context. These context categories include (1) eight types of locations (office, home, outdoors, indoors, restaurant, car, street, and shop), (2) six ambient audio configurations (no speech, user speaking, other speaking, distant voice, loud crowd, and laughter), (3) seven postures (unknown, lying, sitting, standing, walking, running, and biking), and (4) eight activities (no activity, eating, typing, shaking hands, clapping hands, driving, brushing teeth, and washing the dishes).

The within-category contexts recognized by the four category experts are then combined to determining moments of interest in the wearer's everyday life, which are then recorded as a sort of personal diary. The context estimates can also be used to assist a wearer in real time. Several things are worthy of mentioning in the *inSense* system. First, the contexts in the recent past provide information for recognizing the current contexts. This type of information can be utilized by, for instance, hidden Markov models.

Second, the four categories of contexts (postures, ambient audio configurations, postures, and activities) are related. For example, knowing that a user is typing would strongly bias the system to believe that he is both sitting and in his office / home, while knowing that a user is in his office / home only weakly hints that he is typing. Since the *inSense* system does not have a GPS, most of the user's location contexts are inferred from the other three categories of contexts.

Third, the relation between these four categories is too complex to be specified manually. The contexts from one category can be combined freely with those from another category, and there are as many as $8 \times 6 \times 7 \times 8 = 2688$ number of combined states. As a result, a good algorithm for this system must be able to explore and exploit the relations among different categories of contexts automatically while avoiding consideration of the exponential number of combined states.

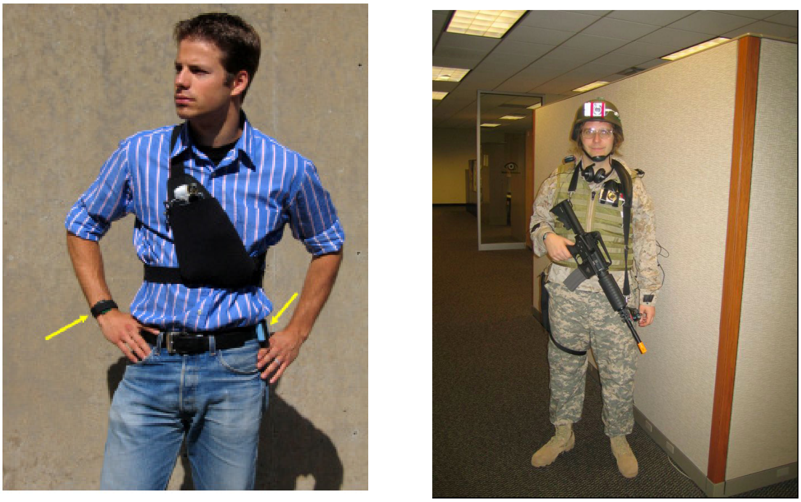

Fig. 1. Left: the *inSense* system; Right: the *ASSIST* system

A very similar approach was taken in developing a soldier state recognition algorithm for the *ASSIST* system (see Figure 1), in collaboration with IBM and Georgia Tech teams. In this system, data was collected in real-time from several accelerometers, microphones, cameras, and a GPS/altimeter, all attached to different parts of the soldiers' clothing. Inference of soldier state was made in real-time, and data automatically shared among different soldiers wearing the *ASSIST* systems based on the pattern of activity shown among the group of soldiers.

Both systems used the same team-of-experts approach. In the *inSense* system, the raw data are computed upon in fundamentally different ways to get the four categories of contexts. The understandings in different categories of contexts can serve to enhance each other. In addition, there are so many combinations of contexts that the curse of dimensionality should be carefully avoided. In comparison, the sample sequences from different sensors in the *ASSIST* system observe very different probability laws and can hardly be jointly Gaussian, thus it was beneficial to apply different models to different sensors and then combine the results.

Since sensor failures were unavoidable due to insufficient power supply, sensor faults, connection errors, or other unpredictable causes, the team-of-experts approach allowed us to offset deficiencies in sensor data by providing estimates of the missing data that were constructed from the other experts' results.

A second example illustrates our approach from the problem of combining several smart sensors on a single individual to the problem of modeling a group of persons over an extended time. The modes of input for this multi-person experiment included not only location sensor data for an individual, but also the sensor data on how the individuals interact with each other.

In our Reality Mining project [5], 81 participants wore Nokia 6600 mobile phones with a custom version of the Context software [9] for a period of over nine months, and have their locations (in terms of cell tower usages), proximity information (in terms of the Bluetooth devices seen), cell phone usages (phone calls, short messages, voice mails), and cell phone states continuously collected. From the recorded data set, we are able to infer the participants' social circles, as well as their individual behaviors. As are typical in such large, extended experiments, the cell phone data collection for the individuals experienced several different types of abnormalities, and the cell phone data for different individuals are correlated. Thus this data set provided a good test for our ability to data mine structural relationships among different participants.

The remainder of this article is organized in the following way. In section 2, we formulate the latent structure influence model and give its latent state inference algorithm and its parameter estimation algorithm. In 3 we discuss how the algorithm performs on these example applications, and compare the robustness, accuracy, and efficiency of the method to some other standard approaches to this problem.

2 Influence Modeling of Multi-sensor Dynamics

The influence model is a tractable approximation of the intractable hidden Markov modeling of multiple interacting dynamic processes. While the number of states for the hidden Markov model is the multiplication of the number of states for individual processes, the number of states for the corresponding influence model is the summation of the number of states for individual processes. The influence model attains this tractability by linearly combining the contributions of latent state distributions of individual processes at time t to get the latent state distributions of individual processes at time $t+1$.

In the rest of this section, we describe the influence parameters, the evolution of the marginal latent state distributions for individual processes, and the observations for individual processes as probabilistic functions of the latent states. The usage with an influence model is generally: inference of latent states given parameters and observations, estimation of parameters given latent states and observations, or simultaneous latent state inference and parameter estimation from observations. A graphical model representation of the influence model is plotted in Figure 2.

A latent structure influence process $\{s_t^{(c)}, y_t^{(c)} : c = \{1, \cdots, C\}, t \in \mathbb{N}\}$ is a stochastic process composed of C interacting (sub-) processes. Each process $c \in \{1, \cdots, C\}$ has latent states $\{s_t^{(c)} : t \in \mathbb{N}\}$ and observations $\{y_t^{(c)} : t \in \mathbb{N}\}$ corresponding to sample times $t \in \mathbb{N}$. The latent structure influence process is normally used to estimate the latent states and/or the parameters based on the observations.

The latent state $s_t^{(c)}$ for processes c at time t is a random variable valued over $\{1, \cdots, m_c\}$. The latent state for all C processes at time t is thus $\boldsymbol{s}_t = (s_t^{(1)}, \cdots, s_t^{(C)})$. We write the probability measures of $s_t^{(c)}$ over its values into a row vector $\boldsymbol{p}(s_t^{(c)})$ and concatenate these row vectors for all processes $c \in \{1, \cdots, C\}$ into a longer row vector $\boldsymbol{p}(\boldsymbol{s}_t)$,

$$\begin{aligned}
\boldsymbol{p}(s_t^{(c)}) &\triangleq \Big(\Pr(s_t^{(c)} = 1), \ldots, \Pr(s_t^{(c)} = m_c)\Big) \\
\boldsymbol{p}(\boldsymbol{s}_t) &\triangleq \Big(\boldsymbol{p}(s_t^{(1)}), \cdots, \boldsymbol{p}(s_t^{(C)})\Big) \\
&= \Big(\Pr(s_t^{(1)} = 1), \ldots, \Pr(s_t^{(1)} = m_1), \cdots, \Pr(s_t^{(C)} = 1), \ldots, \Pr(s_t^{(C)} = m_C)\Big).
\end{aligned}$$

According to the definition of the probability measure, $\sum_{j=1}^{m_c} \Pr(s_t^{(c)} = j) = 1$ for $c \in \{1, \cdots, C\}$.

The probability distributions $P(s_t^{(c)})$ for latent states $s_t^{(c)}$, where $c \in \{1, \cdots, C\}$ and $t \in \{1, \cdots, T\}$, evolve recursively and linearly in a similar way as in the hidden Markov process case. The (initial) probability measure of the latent state $s_{t=1}^{(c)}$ for process $c \in \{1, \cdots, C\}$ at time $t = 1$ over its values is parameterized as a row vector $\pi^{(c)}$, which in turn, is concatenated into a longer row vector $\boldsymbol{\pi}$.

$$\begin{aligned}
\pi_i^{(c)} &\triangleq \Pr(s_1^{(c)} = i), \text{ where } 1 \le i \le m_c \\
\pi^{(c)} &\triangleq \Big(\pi_1^{(c)}, \cdots, \pi_{m_c}^{(c)}\Big) \\
\boldsymbol{\pi} &\triangleq \Big(\pi^{(1)}, \cdots \pi^{(C)}\Big) \\
&= \Big(\pi_1^{(1)}, \cdots \pi_{m_1}^{(1)}, \cdots, \pi_1^{(C)}, \cdots, \pi_{m_C}^{(C)}\Big)
\end{aligned}$$

The probability measures $\boldsymbol{p}(\boldsymbol{s}_t)$ evolve over time linearly as $\boldsymbol{p}(\boldsymbol{s}_{t+1}) = \boldsymbol{p}(\boldsymbol{s}_t) \cdot H$, where H is called an *influence matrix*, as compared to a Markov matrix in a hidden Markov model. The influence matrix H is parameterized in accordance with Asavathiratham's initial parameterization [1]: Call the $C \times C$ matrix $D_{C\times C}$, whose columns each add up to 1, as a *network (influence) matrix*; Call the $m_{c_1} \times m_{c_2}$ Markov matrices $A^{(c_1,c_2)}$ (where $c_1, c_2 \in \{1, \cdots, C\}$), whose rows each add up to 1, as *inter-process state transition matrices*. The influence matrix is formed as the generalized Kronecker product

$$H \triangleq D \otimes \Big\{A^{(c_1,c_2)}\Big\}_{c_1,c_2 \in \{1,\cdots,C\}} = \Big(d_{c_1,c_2} A^{(c_1,c_2)}\Big)_{c_1,c_2 \in \{1,\cdots,C\}},$$

which is a block matrix, whose submatrix at row c_1 and column c_2 is $d_{c_1,c_2} A^{(c_1,c_2)}$, and whose element indexed by (c_1, c_2, i, j) is $h_{i,j}^{(c_1,c_2)} = d_{c_1,c_2} \cdot a_{i,j}^{(c_1,c_2)}$. The fact that $\boldsymbol{p}(\boldsymbol{s}_t)$ is a concatenation of probability distributions is guaranteed by the requirement that each column of D, as well as each row of $A^{(c_1,c_2)}$, adds up to 1.

Using this notation, the latent state distributions $\boldsymbol{p}(\boldsymbol{s}_t)$ for the C interacting processes are evolved as

$$\boldsymbol{p}(\boldsymbol{s}_1) = \boldsymbol{\pi}$$
$$\boldsymbol{p}(\boldsymbol{s}_{t+1}) = \boldsymbol{p}(\boldsymbol{s}_t) \cdot H$$

An observation $y_t^{(c)}$ for process c at sample time t is a random variable conditioned on the corresponding latent state $s_t^{(c)}$. The observations are used to adjust the estimation of latent states

$$P(\boldsymbol{s}_t) \cdot P(\boldsymbol{y}_t | \boldsymbol{s}_t) \triangleq \prod_{c=1}^{C} P(s_t^{(c)}) \cdot P(y_t^{(c)} | s_t^{(c)}).$$

When the observation $y_t^{(c)}$ is finite valued, $y_t^{(c)} \in \{1, \ldots, n_c\}$, we write $b_{i,j}^{(c)} = \Pr(y^{(c)} = j | s^{(c)} = i)$ and call the $m_c \times n_c$ matrix $B^{(c)} = (b_{i,j}^{(c)})$ as an *observation matrix*. When the observation $y_t^{(c)}$ is in multivariate normal distribution, we use n_c to represent the dimensionality of $y_t^{(c)}$, and use $\boldsymbol{\mu}_{s^{(c)}}^{(c)}$, $\Sigma_{s^{(c)}}^{(c)}$ to represent the mean and variance of $y_t^{(c)}$. In other words, when the corresponding latent state is valued as $s_t^{(c)}$, we have $y_t^{(c)} \sim N_{n_c}(\mu_{s^{(c)}}^{(c)}, \Sigma_{s^{(c)}}^{(c)})$.

The latent structure influence process $\{s_t^{(c)}, y_t^{(c)} : c = \{1, \cdots, C\}, t \in \mathbb{N}\}$ is a simplification of the hidden Markov process $\{\boldsymbol{s}_t = \left(s_t^{(1)}, \cdots, s_t^{(C)}\right), \boldsymbol{y}_t = \left(y_t^{(1)}, \cdots, y_t^{(C)}\right) : t \in \mathbb{N}\}$. In the hidden Markov process $\{\boldsymbol{s}_t, \boldsymbol{y}_t : t \in \mathbb{N}\}$, a latent state $\boldsymbol{s}_t$ can take $\prod_{c=1}^{C} m_c$ number of values, an observation $\boldsymbol{y}_t$ can observe very complex distributions conditioned on $\boldsymbol{s}_t$, and the state transition matrix is a $(\prod_c m_c) \times (\prod_c m_c)$ Markov matrix. When C is large, the computation on $\{\boldsymbol{s}_t, \boldsymbol{y}_t : t \in \mathbb{N}\}$ becomes intractable, and is easy to overfit. In comparison, in the latent structure influence process, we only need to cope with the marginal probability distributions $\Pr(s_t^{(c)} = i)$ for state $\boldsymbol{s}_t$, and can cope with $y_t^{(c)}$ for individual interacting processes $c \in \{1, \cdots, C\}$ separately. Asavathiratham [1] proved the following theorems concerning the relationship between an influence process and a Markov process: (1) Given any influence process $\{s_t^{(c)} : c \in \{1, \cdots, C\}, t \in \mathbb{N}\}$ parameterized by the initial state distributions $\boldsymbol{\pi} = \boldsymbol{p}(\boldsymbol{s}_1)$ and the influence matrix H, there exists a Markov process $\{\boldsymbol{x}_t = (x_t^{(1)}, \cdots, x_t^{(C)}) : t \in \mathbb{N}\}$ parameterized by the initial state distribution of $\boldsymbol{x}_1$ and the $(\prod_c m_c) \times (\prod_c m_c)$ Markov matrix G, and the corresponding influence process $\{x_t^{(c)} : c \in \{1, \cdots, C\}, t \in \mathbb{N}\}$ has the same probability measure as the original influence process $\{s_t^{(c)} : c \in \{1, \cdots, C\}, t \in \mathbb{N}\}$ (i.e., both influence processes have the same parameters). (2) Given any Markov process $\{\boldsymbol{x}_t = (x_t^{(1)}, \cdots, x_t^{(C)}) : t \in \mathbb{N}\}$ with Markov matrix G, the stochastic process $\{x_t^{(c)} : c \in \{1, \cdots, C\}, t \in \mathbb{N}\}$ is an influence process with influence matrix H. The two matrices are connected by an *event matrix* $B(m_1, \cdots, m_C)$ (where B is determined only by $m_1, \cdots, m_C$), $B \cdot H = G \cdot B$. As a result, the stationary distribution of the Markov process can be linearly mapped into the stationary distribution of the corresponding influence process. We extended Asavathiratham's

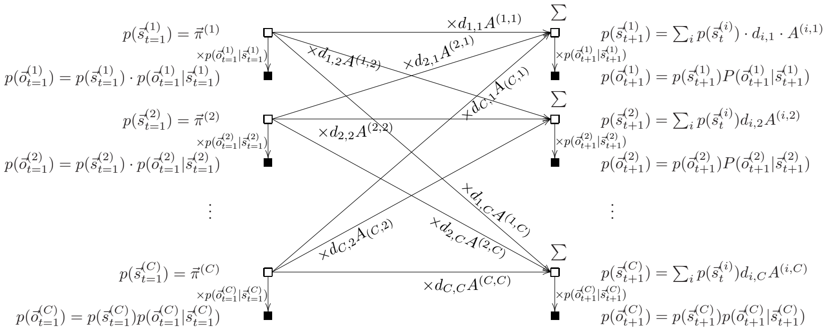
$\sum$
$\times d_{1,1}A^{(1,1)}$
$p(\vec{s}^{(1)}_{t=1}) = \vec{\pi}^{(1)}$
$p(\vec{s}^{(1)}_{t+1}) = \sum_i p(\vec{s}^{(i)}_t) \cdot d_{i,1} \cdot A^{(i,1)}$
$\times d_{1,2}A^{(1,2)}$
$\times d_{2,1}A^{(2,1)}$
$\times p(\vec{o}^{(1)}_{t=1}|\vec{s}^{(1)}_{t=1})$
$\times p(\vec{o}^{(1)}_{t+1}|\vec{s}^{(1)}_{t+1})$
$p(\vec{o}^{(1)}_{t=1}) = p(\vec{s}^{(1)}_{t=1}) \cdot p(\vec{o}^{(1)}_{t=1}|\vec{s}^{(1)}_{t=1})$
$p(\vec{o}^{(1)}_{t+1}) = p(\vec{s}^{(1)}_{t+1})P(\vec{o}^{(1)}_{t+1}|\vec{s}^{(1)}_{t+1})$
$\times d_{C,1}A^{(C,1)}$
$\sum$
$p(\vec{s}^{(2)}_{t=1}) = \vec{\pi}^{(2)}$
$p(\vec{s}^{(2)}_{t+1}) = \sum_i p(\vec{s}^{(i)}_t)d_{i,2}A^{(i,2)}$
$\times d_{2,2}A^{(2,2)}$
$\times p(\vec{o}^{(2)}_{t=1}|\vec{s}^{(2)}_{t=1})$
$\times p(\vec{o}^{(2)}_{t+1}|\vec{s}^{(2)}_{t+1})$
$p(\vec{o}^{(2)}_{t=1}) = p(\vec{s}^{(2)}_{t=1}) \cdot p(\vec{o}^{(2)}_{t=1}|\vec{s}^{(2)}_{t=1})$
$p(\vec{o}^{(2)}_{t+1}) = p(\vec{o}^{(2)}_{t+1})P(\vec{o}^{(2)}_{t+1}|\vec{s}^{(2)}_{t+1})$
$\vdots$
$\vdots$
$\times d_{1,C}A^{(1,C)}$
$\times d_{C,2}A^{(C,2)}$
$\times d_{2,C}A^{(2,C)}$
$\sum$
$p(\vec{s}^{(C)}_{t=1}) = \vec{\pi}^{(C)}$
$p(\vec{s}^{(C)}_{t+1}) = \sum_i p(\vec{s}^{(i)}_t)d_{i,C}A^{(i,C)}$
$\times d_{C,C}A^{(C,C)}$
$\times p(\vec{o}^{(C)}_{t=1}|\vec{s}^{(C)}_{t=1})$
$\times p(\vec{o}^{(C)}_{t+1}|\vec{s}^{(C)}_{t+1})$
$p(\vec{o}^{(C)}_{t=1}) = p(\vec{s}^{(C)}_{t=1})p(\vec{o}^{(C)}_{t=1}|\vec{s}^{(C)}_{t=1})$
$p(\vec{o}^{(C)}_{t+1}) = p(\vec{s}^{(C)}_{t+1})p(\vec{o}^{(C)}_{t+1}|\vec{s}^{(C)}_{t+1})$

Fig. 2. A graphical model representation of the influence model. The left column represents basis step, and the right column represents the induction step. Shadowed squares are observable, while un-shadowed squares are unobservable. Our task is to learn the parameters and latent states from observations. The two-column convention is adopted from Murphy [6].

influence process $\{s_t^{(c)} : c \in \{1, \cdots, C\}, t \in \mathbb{N}\}$ into a latent structure influence process $\{s_t^{(c)}, y_t^{(c)} : c \in \{1, \cdots, C\}, t \in \mathbb{N}\}$, and use the latent structure influence process to understand/simulate how a group of experts cooperate with each other and make predictions.

The forward-backward algorithm for latent state estimation and the maximum likelihood algorithm for parameter estimation for an influence model are derived from the equivalence of the influence model and the corresponding hidden Markov model. Being able to model the dynamics of C interacting processes, with m_c number of latent states for individual process c, in a polynomial complexity in the sum of the number of latent states for individual chains $O(\sum m_c)$ does not necessarily guarantee that the latent state estimation and the parameter estimation algorithms also have a polynomial time complexity. We give the latent state estimation (E-step), as well as the parameter estimation algorithm (M-step) in Algorithm 1. The derivation is deferred in the Appendix. In this algorithm, the random variables $y_t^{(c)}$ for $c \in \{1, \cdots, C\}$ and $t \in \{1, \cdots, T\}$ are already sampled and their values are known. We write the probability (or probability density) of observing $y_t^{(c)}$ when the latent state $s_t^{(c)}$ take values $1, \cdots, m_c$ into a $m_c \times 1$ row vector $\boldsymbol{b}_t^{(c)} \triangleq \left(\Pr(y_t^{(c)}|s_t^{(c)} = 1), \cdots, \Pr(y_t^{(c)}|s_t^{(c)} = m_c)\right)$, and concatenate them into a $(\sum_c m_c) \times 1$ row vector $\boldsymbol{b}_t$. The quantities $\alpha_{t,i}^{(c)} \triangleq \Pr(s_t^{(c)} = i|\{y_{t_0}^{(c_0)} : c_0 \in \{1, \cdots, C\}, t_0 \in \{1, \cdots, t\}\})$ are *forward parameters*. We write $\alpha_{t,i}^{(c)}$ for all $i \in \{1, \cdots, m_c\}$ into a $m_c \times 1$ row vector $\boldsymbol{\alpha}_t^{(c)}$, and concatenate $\boldsymbol{\alpha}_t^{(c)}$ into a $(\sum_c m_c) \times 1$ row vector $\boldsymbol{\alpha}_t$. The quantities $\beta_{t,i}^{(c)} \triangleq \Pr(\{y_{t_0}^{(c_0)} : c_0 \in \{1, \cdots, C\}, t_0 \in \{t+1, \cdots, T\}\}|s_t^{(c)} = i)$ are *backward parameters*. We write $\beta_{t,i}^{(c)}$ for all $i \in \{1, \cdots, m_c\}$ into a $1 \times m_c$ column vector

$\boldsymbol{\beta}_t^{(c)}$, and concatenate $\boldsymbol{\beta}_t^{(c)}$ into a $1 \times (\sum_c m_c)$ column vector $\boldsymbol{\beta}_t$. The quantities $\gamma_{t,i}^{(c)} \triangleq \Pr(s_t^{(c)} = i | \{y_{t_0}^{(c_0)} : c_0 \in \{1, \cdots, C\}, t_0 \in \{1, \cdots, T\}\})$ are *one-slice parameters.* We write $\gamma_{t,i}^{(c)}$ for all $i \in \{1, \cdots, m_c\}$ into a $m_c \times 1$ row vector $\boldsymbol{\gamma}_t^{(c)}$, and concatenate $\boldsymbol{\gamma}_t^{(c)}$ into a $(\sum_c m_c) \times 1$ row vector $\boldsymbol{\gamma}_t$. The quantities $\xi_{t-1 \to t, i \to j}^{(c_1, c_2)} \triangleq \Pr(s_{t-1}^{(c_1)} = i, s_t^{(c_2)} = j | \{y_{t_0}^{(c_0)} : c_0 \in \{1, \cdots, C\}, t_0 \in \{1, \cdots, T\}\})$ are *two-slice parameters.* We write $\xi_{t-1 \to t, i \to j}^{(c_1, c_2)}$ for all $i \in \{1, \cdots, m_{c_1}\}$ and $j \in \{1, \cdots, m_{c_2}\}$ into a $m_{c_1} \times m_{c_2}$ matrix $\xi_{t-1 \to t}^{(c_1, c_2)}$, and concatenate $\xi_{t-1 \to t}^{(c_1, c_2)}$ into a $(\sum_c m_c) \times (\sum_c m_c)$ matrix whose submatrix at row c_1 and column c_2 is $\xi_{t-1 \to t}^{(c_1, c_2)}$.

3 Experimental Results

In this section, we illustrate how an influence model can capture the correlations among different dynamic classification processes. We will show how capturing the correct structure between different "experts" can allow improvement of the overall classification performance. We will also illustrate the efficiency and robustness to noise that this modeling capability provides.

We begin with a synthetic example of a noisy sensor net in order to illustrate the structure that the influence model tries to capture, and how an influence model can be used to improve classification precision. We then extend the noisy body sensor net example and compare the training errors and the testing errors of different dynamic models.

We will then show application of the algorithm to two real examples:

The first example is a wearable smart sensor net example in which the goal is real-time context recognition, and the influence model is used to discover hidden structure among speech, location, activity, and posture classification experts in order to allow for more accurate and robust classification of the wearer's overall state.

The second example is a group of 81 people carrying smart phones that are programmed to record location, proximity to other experimental subjects, and cell phone usage. In this example we will focus on the ability of the influence model to correctly determine the social structure of the group.

3.1 Combining Evidence with the Influence Model

Central to the latent structure influence model is the mechanism that the evidence is combined in the latent state level over time. This mechanism both enables coping with hetereogeneous types of evidence, and makes it possible to automatically find out relations among the different pieces of evidence.

In this subsection, we use a simple example involving two Gaussian distributions to illustrate how the influence model combine evidence and set priors for individual related processes ("experts"). We also compare the mechanisms of the latent structure influence model, the hidden Markov model with full covariant matrix, and the hidden Markov model with diagonal covariant matrix.

Algorithm 1. The EM algorithm for the latent structure influence model

E-Step

$$\boldsymbol{\alpha}_t^* = \begin{cases} \boldsymbol{\pi}_{1\times\sum m_c} \cdot \mathtt{diag}[\boldsymbol{b}_1] & t=1 \\ \boldsymbol{\alpha}_{t-1} \cdot H \cdot \mathtt{diag}[\boldsymbol{b}_t] & t>1 \end{cases}$$

$$\mathcal{N}_t = \mathtt{diag}\left[\left(\underbrace{\frac{1}{\sum\limits_{i=1}^{m_1} \boldsymbol{\alpha}_{t,i}^{*(1)}}, \cdots, \frac{1}{\sum\limits_{i=1}^{m_1} \boldsymbol{\alpha}_{t,i}^{*(1)}}}_{m_1}, \cdots, \underbrace{\frac{1}{\sum\limits_{i=1}^{m_C} \boldsymbol{\alpha}_{t,i}^{*(C)}}, \cdots, \frac{1}{\sum\limits_{i=1}^{m_C} \boldsymbol{\alpha}_{t,i}^{*(C)}}}_{m_C}\right)\right]$$

$$\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_t^* \cdot \mathcal{N}_t$$

$$\boldsymbol{\beta}_t = \begin{cases} \mathbf{1}_{\sum m_c \times 1} & t=T \\ H \cdot \mathtt{diag}[\boldsymbol{b}_t] \cdot \mathcal{N}_{t+1} \cdot \boldsymbol{\beta}_{t+1} & t<T \end{cases}$$

$$\boldsymbol{\gamma}_t = \boldsymbol{\alpha}_t \cdot \mathtt{diag}[\boldsymbol{\beta}_t]$$

$$\xi_{t-1\to t} = \mathtt{diag}[\boldsymbol{\alpha}_{t-1}] \cdot H \cdot \mathtt{diag}[\boldsymbol{b}_t] \cdot \mathcal{N}_t \cdot \mathtt{diag}[\boldsymbol{\beta}_t]$$

$$p(\boldsymbol{y}) = \prod_{t,c} (\sum_{i=1}^{m_c} \boldsymbol{\alpha}_{t,i}^{*(c)})$$

M-Step

- Parameters related to the latent state transitions

$$A^{(i,j)} = \mathtt{normalize}[\sum_{t=2}^{T} \xi_{t-1\to t}^{(i,j)}]$$

$$S = \begin{pmatrix} \mathbf{1}_{1\times m_1} & & \\ & \ldots & \\ & & \mathbf{1}_{1\times m_C} \end{pmatrix}$$

$$d_{ij} = \mathtt{normalize}[S \left(\sum_{t=2}^{T} \xi_{t-1\to t}\right) S^{\mathtt{T}}]$$

$$\boldsymbol{\pi}^{(c)} = \mathtt{normalize}[\boldsymbol{\gamma}_1^{(c)}]$$

- Parameters related to multinomial observations

$$B^{(c)} = \mathtt{normalize}[\sum_t \boldsymbol{\gamma}_t^{(c)\,\mathtt{T}} \cdot \left(\delta(y_t^{(c)}, 1), \cdots, \delta(y_t^{(c)}, \cdots, n_c)\right)]$$

- Parameters related to Gaussian observations

$$\boldsymbol{\mu}^{(c)} = \left(\sum_t \boldsymbol{\gamma}_t^{(c)\,\mathtt{T}} \cdot \boldsymbol{y}_t^{(c)}\right) / \left(\sum_t \boldsymbol{\gamma}_t^{(c)} \cdot \mathbf{1}_{m_c\times 1}\right)$$

$$\Sigma_i^{(c)} = \left(\sum_t \gamma_{t,i}^{(c)} \boldsymbol{y}_t^{(c)} \boldsymbol{y}_t^{(c)\,\mathtt{T}}\right) / \left(\sum_t \boldsymbol{\gamma}_t^{(c)} \cdot \mathbf{1}_{m_c\times 1}\right) - \boldsymbol{\mu}_i^{(c)} \cdot \boldsymbol{\mu}_i^{(c)\,\mathtt{T}}$$

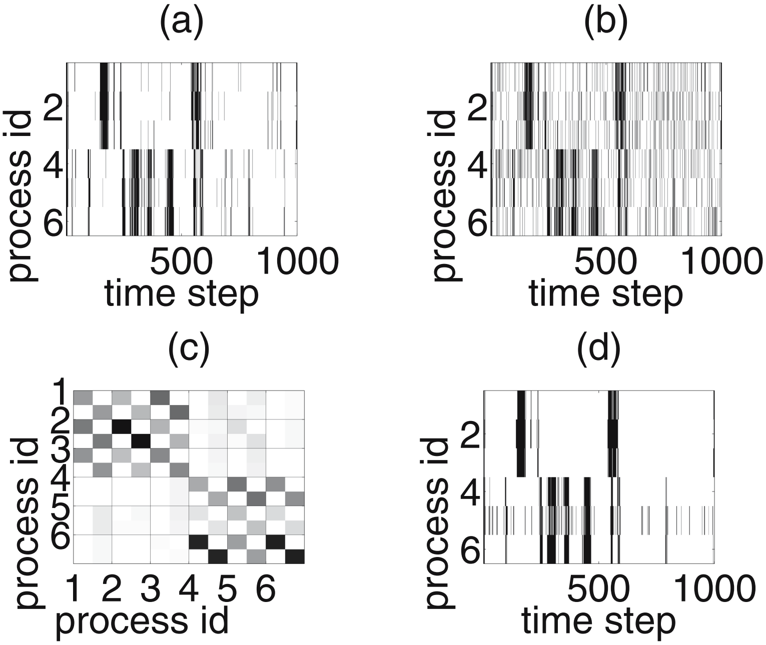

Fig. 3. Inference from observations of interacting dynamic processes

Noisy Body Sensor Net Example. In the noisy body sensor net example, we have six stochastic processes, and we sample these six processes with six body sensors. Each process can be either signaled (one) or non-signaled (zero) at any time, and the corresponding body sensor has approximately 10% of its samples flipped. The interaction of the six stochastic processes behind the scene looks like this: processes one through three tend to have the same states; processes four through six tend to have the same states; the processes are more likely to be non-signaled than to be signaled; and the processes tend to stick to their states for a stretch of time. The parameters of the model are given as the following and are going to be estimated: $A_{ij} = \begin{pmatrix} .99 & .01 \\ .08 & .92 \end{pmatrix}, 1 \le i, j \le 6$, $B_i = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}, 1 \le i \le 6$, $d_{ij} = .33, 1 \le i, j \le 3$, and $d_{ij} = .33, 4 \le i, j \le 6$.

In Figure 3, (a) shows the sampled latent state sequences, (b) shows the corresponding observation sequences, (c) shows the influence matrix reconstructed from sampled observation sequences, and (d) shows the reconstructed latent state sequences after 300 observations. The $(i, j)^{th}$ entry of the $(c_1, c_2)^{th}$ sub-matrix of an influence matrix determines how likely that process c_1 is in state i at time t and process c_2 is in state j at time $t + 1$. It can be seen from Figure 3 (c) that the influence model computation recovers the structure of the interaction.

The influence model can normally attain around 95% accuracy in predicting the latent states for each process. The reconstructed influence matrix has only

9% relative differences with the original one. Using only observations of other chains we can predict a missing chain's state with 87% accuracy.

Comparison of Dynamic Models. The training errors and the testing errors of the coupled hidden Markov model, the hidden Markov model, and the influence model are compared in this example. The setup of the comparison is described as the following. We have a Markov process with 2^C, where $C = 10$, number of states and a randomly generated state transition matrix. Each system state $\boldsymbol{s}_t$ is encoded into a binary $s_t^{(1)} \cdots s_t^{(C)}$. Each of the $m_c = 2$ evaluations of "digit" $s_t^{(c)}$corresponds a different 1-d Gaussian observation $o_t^{(c)}$: Digit $s_t^{(c)} = 1$ corresponds to $o_t^{(c)} \sim \mathcal{N}[\mu_1 = 0, \sigma_1^2 = 1]$; Digit $s_t^{(c)} = 2$ corresponds to $o_t^{(c)} \sim \mathcal{N}[\mu_2 = 1, \sigma_2^2 = 1]$.

In most real sensor nets we normally have redundant measures and an insufficient observations to accurately characterize sensor redundancy using standard methods. Figure 4 compares the performances of several dynamic latent structure models applicable to multi-sensor systems. Of the 1000 samples $(\boldsymbol{o}_t)_{1 \le t \le 100}$, we use the first 250 for training and all 1000 for validation.

There are two interesting points. First, the logarithmically scaled number of parameters of the influence model allows us to attain high accuracy based on a relatively small number of observations. This is because the eigenvectors of the master Markov model we want to approximate are either mapped to the eigenvectors of the corresponding influence model, or mapped to the null space of the corresponding event matrix thus is not observable from the influence model, and

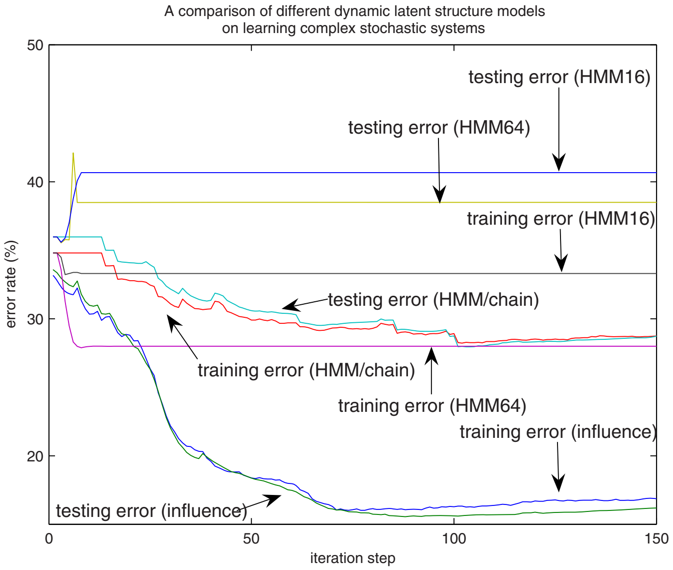

Fig. 4. Comparison of dynamic models

that in addition the eigenvector with the largest eigenvalue (i.e., 1) is mapped to the eigenvector with the largest eigenvalue of the influence matrix [1].

Secondly, both the influence model and the hidden Markov model applied to individual processes are relatively immune to over-fitting, at the cost of low convergence rates. This situation is intuitively the same as the numerical analysis wisdom that a faster algorithm is more likely to converge to a local extremum or to diverge.

3.2 On-Body Smart Sensor Network

In the *inSense* system the sensors consist of a chest-mounted camera, a Wi-Fi transceiver, an ambient audio recorder, and two accelerometers, worn on hip and wrist (see Figure 1) [3]. This system is designed to classify in real time eight locations, six speaking/non-speaking status, seven postures, and eight activities. The classification is carried out in two steps: A pre-classifier (either a single Gaussian, mixture of Gaussians, or C4.5 classifier) is first invoked on the audio and accelerometer features to get a moderately accurate pre-classification result within each the above four categories. These are the “experts” that we desire to group in order to produce more accurate estimates of the wearer’s context.

The pre-classification result of different categories is then fed into an influence model to learn inter-sensor structure, and then this learned structure is used to generate an improved post-classification result. In this example the influence model learns the conditional probabilities that relate the four categories (location, audio, posture, and activity) and then uses this learned influence matrix to improve the overall performance.

For example, given that the *inSense* wearer is typing, we can inspect the row of the influence matrix corresponding to “typing” and see that this person is very likely to be either in the office or at home, to not be speaking, and to be sitting. As a result, the action of typing can play a critical role to disambiguating confusions between sitting and standing, or between speaking vs not-speaking, but not between office and home.

By combining evidence across different categories using the influence model, the classification errors for locations, speaking/non-speaking, postures, and activities decreased by an average of 23%, from 38%, 22%, 8% and 27% to 28%, 19%, 8%, and 17% respectively. The post-classification for postures does not show significant improvement because of two reasons: (1) it is already precise enough considering that we have labeling imprecision in our training data and testing data, and (2) it is the driving force for improving the other categories, and no other categories are more certain than the posture category.

3.3 Social Network Example

This example demonstates reconstructing the social structure of a set of subjects from their cellphone-collected data [5]. In this data 81 subjects wore Bluetooth-enabled mobile telephones that recorded which cell towers were visible to the telephone, thus allowing coarse estimation of the wearers’ location, and which

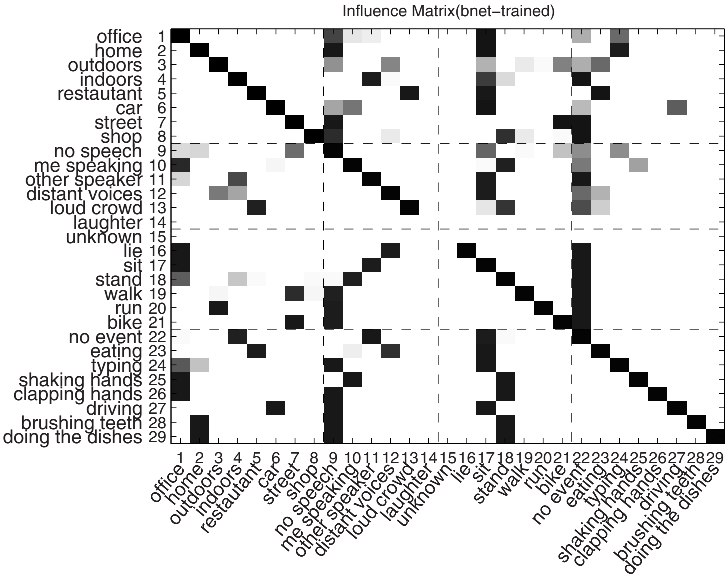

Fig. 5. Influence matrix learned by the EM algorithm

Bluetooth devices are nearby, thus allowing inference of proximity to other subjects. Note that Bluetooth signals include a unique identifier, and are typically detectable at a range of only a few meters. In this study fixed Bluetooth beacons were also employed, allowing fairly precise estimation of subjects location even within buildings. Over the nine months of the study 500,000 hours of data were recorded.

The temporal evolution of these observations were analyzed using the influence model with 81 chains, corresponding to the 81 subjects. Each subjects' chain was constrained to have two latent states ("work", "home") but with no restriction on social network connectivity.

In our first experiment with this data the observation vector for each chain was restricted to the cell tower visibility of each subjects' 10 most commonly seen cell towers. In the resulting model the two states for each subject corresponded accurately to 'in the office' and 'at home', with other locations being misclassified. The resulting influence matrix, shown in Figure 6 (a), demonstrated that most people follow very regular patterns of movement and interpersonal association, and consequently we can predict their actions with substantial accuracy from observations of the other subjects. A few of the chains were highly independent and thus not predictable. These chains corresponded to new students, who had not yet picked up the rhythm of the community, and the faculty advisors, whose patterns are shown to determine the patterns of other students.

In another setup, we used the Bluetooth proximity distribution as our observations. Again, the latent states accurately reflect whether a person is at home

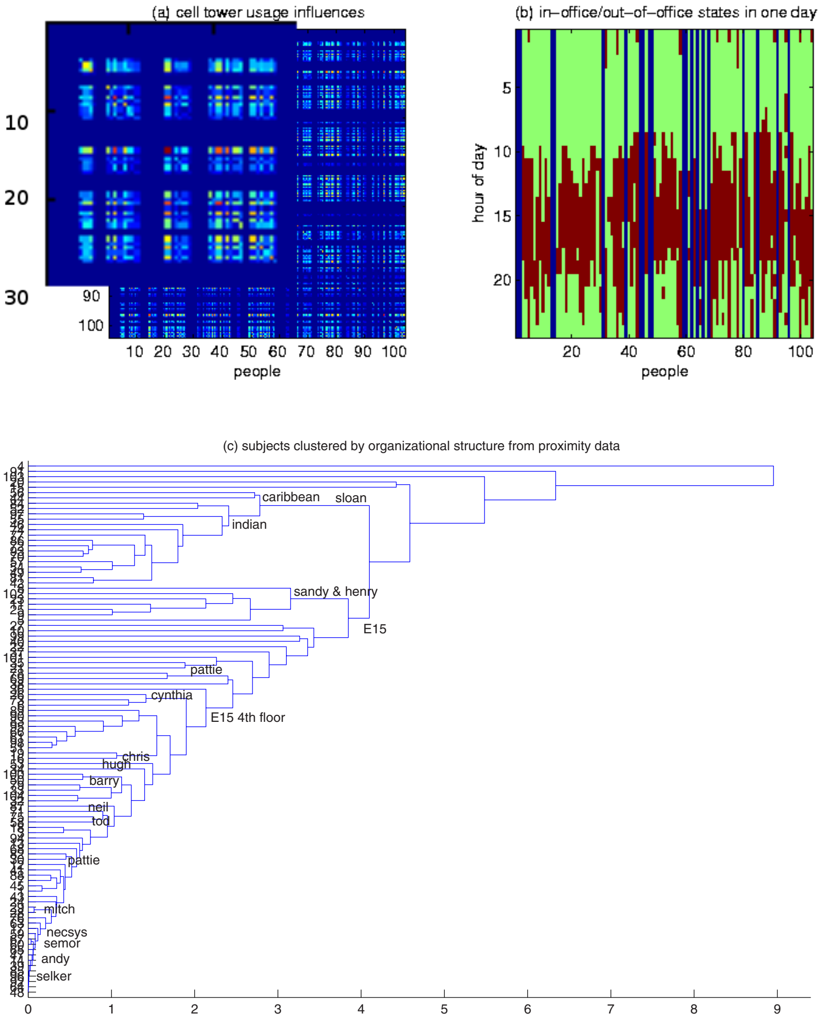

Fig. 6. Finding social structures from cellphone-collected data. (a) New students and faculty are outliers in the influence matrix, appearing as red dots due to large self-influence values. (b) Most People follow regular patterns (red: in office, green: out of office, blue: no data), (c) clustering influence values recovers workgroup affiliation with high accuracy (labels show name of group).

of in office. However with this data the resulting influence matrix shows precisely the social and geometrical structure of the subjects. The dendrogram from the proximity influence matrix shown in Figure 6 (b) captures the actual organization of the laboratory, clustering people into their actual work groups with only three errors. For comparison, a clustering based on direct correlations in the data has six errors.

4 Conclusion

We have presented the formulation of a latent structure influence model, given the parameter learning and latent state estimation algorithms, and demonstrated the latent structure influence model's performance in combining and analyzing networks of experts. Both in simulation and real examples the influence model proved to be an efficient and accurate method of estimating unknown network structure and simultaneously estimating transition parameters. This was shown to allow more accurate estimates of state, and increased tolerance to missing data and similar noise. As a result, we believe that the latent structure influence process will provide a good framework for human computing applications.

Matlab code for the influence model and for the synthetic sensor net example may be found at:
`http://vismod.media.mit.edu/vismod/demos/influence-model/index.html`.

References

[1] Chalee Asavathiratham. *The Influence Model: A Tractable Representation for the Dynamics of Networked Markov Chains*. PhD thesis, MIT, 1996.

[2] Sumit Basu, Tanzeem Choudhury, Brian Clarkson, and Alex Pentland. Learning human interactions with the influence model. Technical report, MIT Media Laboratory Vision & Modeling Technical Report #539, 2001. URL `http://vismod.media.mit.edu/tech-reports/TR-539.pdf`.

[3] Mark Blum, Alex Pentland, and Gehrard Tröster. Insense: Interest-based life logging. In *IEEE Multimedia*, volume 13(4), pages 40–48, 2006.

[4] DARPA. Assist proposer information pamphlet, 2004. URL `http://www.darpa.mil/ipto/solicitations/open/04-38_PIP.htm`.

[5] Nathan Eagle and Alex Pentland. Reality mining: Sensing complex social systems. *Journal of Personal and Ubiquitous Computing*, 2005.

[6] Kevin Murphy. The bayes net toolbox for matlab. In *Computing Science and Statistics*, 2001.

[7] Nuria M. Oliver, Barbara Rosario, and Alex Pentland. A bayesian computer vision system for modeling human interactions. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 22(8), pages 831–843, 2000.

[8] Maja Pantic, Alex Pentland, Anton Nijholt, and Thomas Huang. Human computing and machine understanding of human behavior: A survey. In *Proceedings of the 8th International Converence on Multimodal Interfaces*, pages 239–248, 2006.

[9] M. Raento, A. Oulasvirta, R. Petit, and H. Toivonen. Contextphone — a prototyping platform for context-aware mobile applications. In *IEEE Pervasive Computer*, April 2005.

Appendix: A Derivation of the EM Algorithm for the Influence Process

In section 2, we gave the EM algorithm for the latent structure influence process. The derivation of this algorithm is below.

Latent State Inference

The task of latent state inference is to estimate

$$\left(s_t^{(c)}\right)_{1\le t\le T}^{1\le c\le C} \triangleq \left\{s_t^{(c)} : c \in \{1,\cdots,C\}, t \in \{1,\cdots,T\}\right\}$$

from

$$\left(y_t^{(c)}\right)_{t+1\le t\le T}^{1\le c\le C} \triangleq \left\{y_t^{(c)} : c \in \{1,\cdots,C\}, t \in \{1,\cdots,T\}\right\}$$

given the influence parameters.

Theorem 1. *Let the marginal forward parameters $\alpha_t^{(c)}(s_t^{(c)})$, the marginal backward parameters $\beta_t^{(c)}(s_t^{(c)})$, the marginal one-slice parameters $\gamma_t^{(c)}(s_t^{(c)})$, the marginal two-slice parameters $\xi_{t\to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)})$ of a latent structure influence model be*

$$\begin{aligned}
\alpha_t^{(c)}(s_t^{(c)}) &= P\left(s_t^{(c)}, \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le t}^{1\le c_1\le C}\right)\\
\beta_t^{(c)}(s_t^{(c)}) &= P\left(\left(y_{t_1}^{(c_1)}\right)_{t+1\le t_1\le T}^{1\le c_1\le C} \middle| s_t^{(c)}\right)\\
\gamma_t^{(c)}(s_t^{(c)}) &= P\left(s_t^{(c)} \middle| \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le T}^{1\le c_1\le C}\right)\\
\xi_{t\to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)}) &= P\left(s_t^{(c_1)} s_{t+1}^{(c_2)} \middle| \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le T}^{1\le c_1\le C}\right)
\end{aligned}$$

They can be computed recursively in the following way:

$$\begin{aligned}
\alpha_1^{(c)}(s_1^{(c)}) &= P\left(\left(y_t^{(c)}\right)^{1\le c\le C} | s_1^{(c)}\right) \cdot \pi_{s_1^{(c)}}^{(c)}\\
\alpha_t^{(c)}(s_{2\le t}^{(c)}) &= P\left(\left(y_t^{(c)}\right)^{1\le c\le C} | s_t^{(c)}\right) \sum_{c_1, s_{t-1}^{(c_1)}} \alpha(s_{t-1}^{(c_1)}) h_{s_{t-1}^{(c_1)} s_t^{(c)}}^{(c_1,c)}\\
\beta_T^{(c)}(s_T^{(c)}) &= 1\\
\beta_{t<T}^{(c)}(s_t^{(c)}) &= \frac{1}{C} \cdot \sum_{c_1=1}^{C} \sum_{s_{t+1}^{(c_1)}=1}^{m_{c_1}} h_{s_t^{(c)}, s_{t+1}^{(c_1)}}^{(c,c_1)} \cdot P\left(\left(y_{t+1}^{(c)}\right)^{1\le c\le C} \middle| s_{t+1}^{(c_1)}\right) \beta_{t+1}^{(c)}(s_{t+1}^{(c)})\\
\gamma_t^{(c)}(s_t^{(c)}) &= \alpha_t^{(c)}(s_t^{(c)}) \cdot \beta_t^{(c)}(s_t^{(c)})\\
\xi_{t\to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)}) &= \alpha_t^{(c_1)}(s_t^{(c_1)}) \cdot h_{s_t^{(c_1)}, s_{t+1}^{(c_2)}}^{(c_1,c_2)} \cdot \beta_{t+1}^{(c_2)}(s_{t+1}^{(c_2)}) \cdot P\left(\left(y_{t+1}^{(c)}\right)^{1\le c\le C} \middle| s_{t+1}^{(c_2)}\right)
\end{aligned}$$

Proof. In the following, we demonstrate that we can solve for the marginal forward parameters without first solving the joint marginal forward parameters.

- Basis Step

$$\begin{aligned}
&\alpha(s_1^{(c)})\\
&= P\left(s_t^{(c)}, \left(y_1^{(c_1)}\right)^{1\le c_1\le C}\right)\\
&= P\left(\left(y_1^{(c_1)}\right)^{1\le c_1\le C} | s_1^{(c)}\right)\cdot P\left(s_1^{(c)}\right)\\
&= P\left(\left(y_1^{(c_1)}\right)^{1\le c_1\le C} | s_1^{(c)}\right)\cdot \pi^{(c)}_{s_1^{(c)}}
\end{aligned}$$

- Induction Step

$$\begin{aligned}
&\alpha(s_{t\ge 2}^{(c)})\\
&= P\left(s_t^{(c)}, \left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le t}\right)\\
&= P\left(\left(y_t^{(c_1)}\right)^{1\le c_1\le C} | s_t^{(c)}\right)\cdot P\left(s_t^{(c)}, \left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le t-1}\right)\\
&= P\left(\left(y_t^{(c_1)}\right)^{1\le c_1\le C} | s_t^{(c)}\right)\cdot \sum_{c_1=1}^{C}\sum_{s_{t-1}^{(c_1)}=1}^{m_{c_1}} P\left(s_{t-1}^{(c_1)}, \left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le t-1}\right)\cdot h^{(c_1,c)}_{s_{t-1}^{(c_1)} s_t^{(c)}}\\
&= P\left(\left(y_t^{(c_1)}\right)^{1\le c_1\le C} | s_t^{(c)}\right)\cdot \left(\sum_{c_1=1}^{C}\sum_{s_{t-1}^{(c_1)}=1}^{m_{c_1}} \alpha(s_{t-1}^{(c_1)})\cdot h^{(c_1,c)}_{s_{t-1}^{(c_1)} s_t^{(c)}}\right)
\end{aligned}$$

In the following, we show that we can get the marginal backward parameters without the knowledge of the joint backward parameters.

- Basis Step. We have $\beta(s_T^{(c)}) = 1$ trivially, and

$$\begin{aligned}
\sum_{s_T^{(c)}=1}^{m_c} \alpha(s_T^{(c)})\cdot\beta(s_T^{(c)}) &= \sum_{s_T^{(c)}=1}^{m_c} P\left(s_T^{(c)}, \left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le T}\right)\\
&= P\left(\left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le T}\right)\\
\frac{1}{C}\cdot\sum_{c=1}^{C}\sum_{s_T^{(c)}=1}^{m_c} \alpha(s_T^{(c)})\cdot\beta(s_T^{(c)}) &= P\left(\left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le T}\right)
\end{aligned}$$

– Induction Step

$$
\begin{aligned}
&\beta(s^{(c)}_{t<T})\\
&= P\left(\left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{t+1\leq t_1\leq T}|s^{(c)}_t\right)\\
&= \sum_{s^{(c_1)}_{t+1}=1}^{m_C} P\left(s^{(c_1)}_{t+1}, \left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{t+1\leq t_1\leq T}|s^{(c)}_t\right), 1\leq c_1\leq C\\
&= \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s^{(c_1)}_{t+1}=1}^{m_C} P\left(s^{(c_1)}_{t+1}, \left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{t+1\leq t_1\leq T}|s^{(c)}_t\right)\\
&= \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s^{(c_1)}_{t+1}=1}^{m_C} P\left(\left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{t+1\leq t_1\leq T}|s^{(c_1)}_{t+1}\right)\cdot P(s^{(c_1)}_{t+1}|s^{(c)}_t)\\
&= \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s^{(c_1)}_{t+1}=1}^{m_{c_1}} \beta(s^{(c_1)}_{t+1})\cdot h^{(c_1,c)}_{s^{(c)}_t s^{(c_1)}_{t+1}}\cdot P\left(\left(y^{(c_1)}_{t}\right)^{1\leq c_1\leq C}|s^{(c_1)}_{t+1}\right)
\end{aligned}
$$

The one-slice parameters $\gamma^{(c)}_t(s^{(c)}_t)$ can be computed from the marginal forward parameters and the marginal backward parameters

$$
\begin{aligned}
\gamma^{(c)}_t(s^{(c)}_t) &= P\left(s^{(c)}_t, \left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{1\leq t_1\leq T}\right)\\
&= P\left(s^{(c)}_t, \left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{1\leq t_1\leq t}\right) P\left(\left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{t+1\leq t_1\leq T}|s^{(c)}_t\right)\\
&= \alpha^{(c)}_t(s^{(c)}_t)\cdot\beta^{(c)}_t(s^{(c)}_t)
\end{aligned}
$$

The two-slice parameters $\xi^{(c_1,c_2)}_{t\rightarrow t+1}(s^{(c_1)}_t, s^{(c_2)}_{t+1})$ can also be computed from the marginal forward parameters $\alpha^{(c)}_t(s^{(c)}_t)$ and the marginal backward parameters $\beta^{(c)}_t(s^{(c)}_t)$:

$$
\begin{aligned}
\xi^{(c_1,c_2)}_{t\rightarrow t+1}(s^{(c_1)}_t, s^{(c_2)}_{t+1}) &= P\left(s^{(c_1)}_t s^{(c_2)}_{t+1}, \left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{1\leq t_1\leq T}\right)\\
&= P\left(s^{(c_1)}_t \left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{1\leq t_1\leq t}\right)\cdot P\left(s^{(c_2}_{t+1}|s^{(c_1)}_t\right)\cdot\\
&\quad P\left(\left(y^{(c_1)}_{t+1}\right)^{1\leq c_1\leq C}|s^{(c_2)}_{t+1}\right)\cdot P\left(\left(y^{(c_1)}_{t_1}\right)^{1\leq c_1\leq C}_{t+2\leq t_1\leq T}|s^{(c_2)}_{t+1}\right)\\
&= \alpha^{(c_1)}_t\cdot h^{(c_1,c_2)}_{s^{(c_1)}_t s^{(c_2)}_{t+1}}\cdot P\left(\left(y^{(c_1)}_{t+1}\right)^{1\leq c_1\leq C}|s^{(c_2)}_{t+1}\right)\cdot\beta^{(c_2)}_{t+1}
\end{aligned}
$$

Parameter Estimation

Suppose the latent states at time $t = 1..T$ is already known $\boldsymbol{s}_t = s_t^{(1)} \cdots s_t^{(C)}$. The likelihood function is

$$\begin{aligned}
&P\left(\left(y_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}\right)\\
&= \pi_{\boldsymbol{s}_1} \cdot \left(\prod_{t=1}^{T-1} g_{\boldsymbol{s}_t \to \boldsymbol{s}_{t+1}}\right) \cdot \left(\prod_{t=1}^{T} P(\boldsymbol{y}_t|\boldsymbol{s}_t)\right)\\
&= \left(\prod_{c=1}^{C} \pi^{(c)}_{s_1^{(c)}}\right) \cdot \left(\prod_{t=1}^{T}\prod_{c2=1}^{C}\sum_{c1=1}^{C} h^{(c1,c2)}_{s_t^{(c1)},s_{t+1}^{(c2)}}\right) \cdot \left(\prod_{t=1}^{T}\prod_{c=1}^{C} P(y_t^{(c)}|s_t^{(c)})\right)
\end{aligned}$$

We can find new parameters and try to maximize the log likelihood function:

$$\begin{aligned}
&\log P\left(\left(y_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}\right)\\
&= \sum_{c=1}^{C}\log \pi^{(c)}_{s_1^{(c)}} + \sum_{t=1}^{T}\sum_{c=1}^{C}\log P(y_t^{(c)}|s_t^{(c)}) + \sum_{t=1}^{T}\sum_{c2=1}^{C}\log\sum_{c1=1}^{C} h^{(c1,c2)}_{s_t^{(c1)},s_{t+1}^{(c2)}}
\end{aligned}$$

$$\geq \sum_{c=1}^{C}\log \pi^{(c)}_{s_1^{(c)}} + \sum_{t=1}^{T}\sum_{c=1}^{C}\log P(y_t^{(c)}|s_t^{(c)}) + \sum_{t=1}^{T}\sum_{c2=1}^{C}\sum_{c1=1}^{C}\log h^{(c1,c2)}_{s_t^{(c1)},s_{t+1}^{(c2)}} \tag{1}$$

$$\begin{aligned}
&= \sum_{c=1}^{C}\sum_{i=1}^{m_c}\delta(s_1^{(c)},i)\cdot\log\pi_i^{(c)} + \sum_{t=1}^{T}\sum_{c=1}^{C}\sum_{i=1}^{m_c}\delta(s_t^{(c)},i)\cdot\log P(y_t^{(c)}|i) +\\
&\sum_{t=1}^{T-1}\sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}}\delta(s_t^{(c1)},i)\cdot\delta(s_{t+1}^{(c2)},j)\cdot\log h_{i,j}^{(c1,c2)}
\end{aligned} \tag{2}$$

$$\begin{aligned}
&\triangleq \sum_{c=1}^{C}\sum_{i=1}^{m_c}\tilde{\pi}_i^{(c)}\cdot\log\pi_i^{(c)} + \sum_{c=1}^{C}\sum_{i=1}^{m_c}\tilde{\gamma}^{(c)}(i)\cdot\log P(y_t^{(c)}|i) +\\
&\sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}}\tilde{\xi}^{(c1,c2)}(i,j)\cdot\log h_{i,j}^{(c1,c2)}
\end{aligned}$$

where the step 1 is according to the Jensen's inequality, and the function $\delta(i,j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$ is the Kronecker delta function. From 2, we know that $\tilde{\pi}_i^{(c)} = \delta(s_1^{(c)}, i)$, $\tilde{\xi}^{(c1,c2)}(i,j) = \sum_{t=1}^{T-1}\delta(s_t^{(c1)},i)\cdot\delta(s_{t+1}^{(c2)},j)$, and $\tilde{\gamma}^{(c)}(i) = \sum_{t=1}^{T}\delta(s_t^{(c)},i)$ are the sufficient statistics for $\pi_i^{(c)}$, $h_{i,j}^{(c1,c2)}$, and $P(y_t^{(c)}|i)$ respectively. We can maximize the parameters involved in the influence matrix H by equaling them to the corresponding sufficient statistics:

$$\pi_i^{(c)} = \tilde{\pi}_i^{(c)} \tag{3}$$

$$h_{i,j}^{(c1,c2)} = \frac{1}{C}\cdot\frac{\tilde{\xi}_{i,j}^{(c1,c2)}}{\sum_{j=1}^{m_{c2}}\tilde{\xi}_{i,j}^{(c1,c2)}} \tag{4}$$

We can maximize the parameters that map the latent states to the observations in the same way as in an ordinary hidden Markov model.

When the latent states at time $t = 1..T$ are not known. We can choose parameters that maximize the expected log likelihood function:

$$\begin{aligned}
&\mathtt{E}_{\boldsymbol{s}_1\cdots\boldsymbol{s}_T}\left[\log p\left(\left(y_t^{(c)}\right)_{1\le t\le T}^{1\le c\le C}\right)\right]\\
&=\mathtt{E}_{\boldsymbol{s}_1\cdots\boldsymbol{s}_T}\left[\sum_{c=1}^{C}\log\pi^{(c)}_{s_1^{(c)}}+\sum_{t=1}^{T}\sum_{c2=1}^{C}\log\sum_{c1=1}^{C}h^{(c1,c2)}_{s_t^{(c1)},s_{t+1}^{(c2)}}+\sum_{t=1}^{T}\sum_{c=1}^{C}\log P(y_t^{(c)}|s_t^{(c)})\right]\\
&\ge\mathtt{E}_{\boldsymbol{s}_1\cdots\boldsymbol{s}_T}\left[\sum_{c=1}^{C}\log\pi^{(c)}_{s_1^{(c)}}+\sum_{t=1}^{T}\sum_{c2=1}^{C}\sum_{c1=1}^{C}\log h^{(c1,c2)}_{s_t^{(c1)},s_{t+1}^{(c2)}}+\sum_{t=1}^{T}\sum_{c=1}^{C}\log P(y_t^{(c)}|s_t^{(c)})\right]\\
&=\sum_{c=1}^{C}\sum_{i=1}^{m_c}\mathtt{E}_{s_1^{(c)}}\left[\delta(s_1^{(c)},i)\right]\cdot\log\pi_i^{(c)}+\\
&\quad\sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}}\sum_{t=1}^{T-1}\mathtt{E}_{s_t^{(c1)}s_{t+1}^{(c2)}}\left[\delta(s_t^{(c1)},i)\cdot\delta(s_{t+1}^{(c2)},j)\right]\cdot\log h_{i,j}^{(c1,c2)}+\\
&\quad\sum_{c=1}^{C}\sum_{i=1}^{m_c}\sum_{t=1}^{T}\mathtt{E}_{s_t^{(c)}}\left[\delta(s_t^{(c)},i)\right]\cdot\log P(y_t^{(c)}|i)\\
&\triangleq\sum_{c=1}^{C}\sum_{i=1}^{m_c}\tilde{\pi}_i^{(c)}\cdot\log\pi_i^{(c)}+\\
&\quad\sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}}\tilde{\xi}^{(c1,c2)}(i,j)\cdot\log h_{i,j}^{(c1,c2)}+\sum_{c=1}^{C}\sum_{i=1}^{m_c}\tilde{\gamma}^{(c)}(i)\cdot\log P(y_t^{(c)}|i)
\end{aligned}$$

According to the attributes of the expectation operator and the Kronecker delta operator, the sufficient statistics are given in the following way, and the parameters related to the state transitions are maximized by Equations 3 and 4:

$$\begin{aligned}
\tilde{\pi}_i^{(c)}&=\gamma_i^{(c)}\\
\tilde{\xi}^{(c1,c2)}(i,j)&=\sum_{t=1}^{T=1}\xi_{t\to t+1}^{(c1,c2)}(i,j)\\
\tilde{\gamma}^{(c)}(i)&=\sum_{t=1}^{T}\gamma_t^{(c)}(i)
\end{aligned}$$

The parameters are re-estimated in the same way as in the known latent state case.