Multi-Task Learning for Analyzing and Sorting Large Databases of Sequential Data

Kai Ni, John Paisley, Lawrence Carin and David Dunson[†]

Department of Electrical and Computer Engineering

Institute of Statistics and Decision Sciences[†]

Duke University, Durham, NC 27708 USA

Email: {kn6,jwp4,lcarin}@ee.duke.edu, dunson@stat.duke.edu

Abstract

A new hierarchical nonparametric Bayesian framework is proposed for the problem of multi-task learning (MTL) with sequential data. The models for multiple tasks, each characterized by sequential data, are learned jointly, and the inter-task relationships are obtained simultaneously. This MTL setting is used to analyze and sort large databases composed of sequential data, such as music clips. Within each data set, we represent the sequential data with an infinite hidden Markov model (iHMM), avoiding the problem of model selection (selecting a number of states). Across the data sets, the multiple iHMMs are learned jointly in a MTL setting, employing a nested Dirichlet process (nDP). The nDP-iHMM MTL method allows simultaneous task-level and data-level clustering, with which the individual iHMMs are enhanced and the between-task similarities are learned. Therefore, in addition to improved learning of each of the models via appropriate data sharing, the learned sharing mechanisms are used to infer inter-data relationships of interest for data search. Specifically, the MTL-learned task-level sharing mechanisms are used to define the affinity matrix in a graph-diffusion sorting framework. To speed up the MCMC inference for large databases, the nDP-iHMM is truncated to yield a nested Dirichlet-distribution based HMM representation, which accommodates fast variational Bayesian (VB) analysis for large-scale inference, and the effectiveness of the framework is demonstrated using a database composed of 2500 digital music pieces.

Index Terms

Infinite hidden Markov model, Hierarchical Bayesian modeling, Variational Bayesian, Multi-task Learning

I. INTRODUCTION

Multi-task learning (MTL) [1] has attracted significant interest in the machine learning community over the last decade [2], [3], [4], [5]. Much of the research on MTL has exploited ideas in Bayesian hierarchical modeling [6], and such techniques have been successfully applied to information retrieval [2] and computer vision [3]. In most previous MTL studies, a focus has been placed on data-level clustering; for example, the hierarchical Dirichlet process (HDP) [7] may be used to jointly learn mixture models for multiple tasks, and data are shared between tasks as appropriate (sharing is at the data level, not at the overall task level). However, one may also be interested in task-level clustering, i.e., to learn overall task-level similarities (for example, to learn the degree to which multiple data sets are related to one another). Such inter-data-set relationships may be inferred by such models as the HDP, via measures [8] applied after the individual models are learned. However, such post-processing model-similarity measures are often expensive to implement. The nested Dirichlet process (nDP) [9] has been developed as a hierarchical Bayesian tool for MTL and inter-task sharing, and this framework is extended in the work presented here.

While most work in hierarchical Bayesian modeling addresses clustering multiple exchangeable data sets, little has been done to solve the MTL problem for sequential data. Hidden Markov models (HMMs) have been widely used to analyze sequential data, addressing problems in speech recognition [10], music analysis [8] and multi-aspect target detection [11], among many others. In many scenarios, one may have limited sequential data for training. Rather than building HMMs for each task separately, it is desirable to identify relationships between tasks and share information appropriately, thus obtaining more accurate task-dependent models. In the context of HMM-based sequential-data analysis, a key issue involves development of a methodology for finding the appropriate model complexity, i.e., defining the appropriate number of HMM states. However, the data may not be represented by a single “correct” HMM structure, i.e., a fixed number of states. Rather than performing model selection [12] to select a fixed model structure, we employ a nonparametric Bayesian approach [7] in which the number of states is not fixed *a priori*; the model is termed an infinite HMM (iHMM). To learn multiple iHMMs simultaneously, one for each sequential data set, the base distributions of the iHMMs may be drawn from an nDP [9], this allowing inter-task clustering.

The nDP-iHMM introduced here represents a new hierarchical nonparametric Bayesian model for multi-task learning with sequential data, allowing simultaneous iHMM task-level clustering and data-level modeling. A Markov chain Monte Carlo (MCMC) inference engine has been adopted for the nDP-iHMM. However, such MCMC inference is impractical computationally when considering a large number of tasks. Therefore, another contribution of this paper is the development of a variational Bayesian (VB) [13] inference formulation, which allows relatively fast computation even for analysis of large-scale problems. It requires replacing a truncated iHMM with a Dirichlet-distribution based HMM, which we call a nested Dirichlet distribution HMM (nDD-HMM).

As demonstrated here, the proposed sequential-data MTL setting scales well, with example results presented on a large digital-music database. Specifically, the nDD-HMM algorithm is employed to simultaneously learn models for a large number of sequential data sets; in addition to learning the individual models, a by-product of the analysis is an inter-task affinity matrix, learned via the nDD-HMM sharing mechanisms. This matrix is subsequently normalized to define a random walk on a graph, with graph-diffusion analysis [14] then performed to rank and sort the sequential data. Example results are presented for a database of 2500 pieces of digital music.

As indicated above, the proposed framework relies on a hierarchical Bayesian analysis, with this of particular relevance for the HMM statistical representation of the time-evolving data. The remainder of the paper will focus on this setting, and will cite related work of this type. However, before proceeding, it is worth noting that multi-task learning is a field of significant interest, and other techniques have been pursued, apart from a hierarchical statistical analysis. Although this previous research was not applied previously to time-evolving data, it is of interest to note these other techniques for multi-task learning; the interested reader is referred to [15], [16], [17], [18], [19].

The remainder of the paper is organized as follows. In Section II we briefly review the HDP construction of the iHMM. We introduce the nDP-iHMM framework in Section III, to solve MTL for sequential data. The truncated form of the nDP-iHMM, the nDD-HMM, and its VB inference algorithm are described in Section IV. Experimental results for both synthetic data and a large set of digital music are presented in Section V, with the learned inter-task sharing mechanism feeding into a graph-diffusion-based data-search framework. Conclusions are provided in Section

VI.

II. THE INFINITE HIDDEN MARKOV MODEL FOR A SINGLE TASK

Hidden Markov models (HMMs) have been used extensively for modeling sequential data. A data sequence of length T generated by an HMM yields a sequence of observations $\mathbf{o} = \{o_1, o_2, \dots, o_T\}$ and a sequence of hidden underlying states $\mathbf{s} = \{s_1, s_2, \dots, s_T\}$, the latter following a Markov process. Consider an HMM with N states and M possible observations, for which the parameters of the model are $(\pi^N, A^{N \times N}, B^{N \times M})$, with π being the initial-state probability, A the transition matrix representing $P(s_t | s_{t-1})$, and B the observation matrix representing $P(o_t | s_t)$. HMMs are typically learned via the expectation-maximization (EM) method, implemented by the Baum-Welch algorithm [10] or the variational Bayesian method [13]. However, in both approaches the model structure must be specified initially, i.e. the number of states is fixed. Learning the correct model complexity requires expensive model selection, and in some applications there may exist no such fixed “correct” model (the limited sequential data for a given problem may be best represented via an ensemble of HMMs, with different numbers of states). Here we address this problem by considering an HMM with a countably infinite state space, namely the *infinite hidden Markov model* (iHMM). Beal et al. [20] first proposed the iHMM and later Teh et al. [7] demonstrated that the iHMM can be recast in the hierarchical Dirichlet process (HDP) framework. Below we start with a finite-state HMM, and then extend this to an infinite state space, and we describe how the HDP may be used to constitute the iHMM.

An N -state HMM is a set of N coupled mixture models, each with N mixture components that are state-dependent observation models. Given the previous hidden state $s_{t-1} = n$, the conditional distribution of the next observation o_t is

$$p(o_t | s_{t-1} = n) = \sum_{k=1}^N w_k^n F(o_t | \theta_k^*), \quad (1)$$

where $w_k^n = p(s_t = k | s_{t-1} = n)$ is the element in the transition matrix A , representing the probability of transitioning from the n^{th} state to the k^{th} state and $F(\cdot | \theta_k^*)$ is the k^{th} state-dependent observation model. Therefore, the previous state s_{t-1} indexes a specific row of the transition matrix serving as the mixture weights for choosing current state s_t , and the state-dependent observation models serve as the mixture components generating o_t . Note that for

different possible visited states at time $t-1$ (different states s_{t-1}), the mixture models generating o_t share the same set of mixture components but with different mixture weights.

When considering an infinite state space, the aforementioned mixture models become infinite mixture models, and it is natural to use *Dirichlet process mixture models* [21] to describe such mixture models with unbounded components. Assume the n^{th} infinite mixture model in the iHMM G^n is drawn from a *Dirichlet process* [22] $G^n \sim \text{DP}(\alpha, G^0)$, where α is a positive real number and G^0 is the base distribution for G^n , Ferguson [22] showed that G^n is discrete with probability one, a property made explicit by the stick-breaking construction [23]

$$G^n = \sum_{k=1}^{\infty} w_k^n \delta_{\theta_k^*} \quad w_k^n = \beta_k^{n'} \prod_{l=1}^{k-1} (1 - \beta_l^{n'}) \quad \beta_k^{n'} \stackrel{iid}{\sim} \text{Beta}(1, \alpha), \quad (2)$$

where $\delta_{\theta_k^*}$ is a discrete measure concentrated at θ_k^* , $\theta_k^* \stackrel{iid}{\sim} G^0$ and the procedure of generating w^n is denoted $w^n \sim \text{Stick}(\alpha)$. Here w^n corresponds to the transition probabilities starting from the n^{th} state and θ_k^* are the parameters of the observation model given allocation to component k . To deal with the infinite countable states, the iHMM takes the form of a *hierarchical Dirichlet process* [7]

$$G^0 \mid \gamma, H \sim \text{DP}(\gamma, H), \quad G^n \mid \alpha, G^0 \stackrel{iid}{\sim} \text{DP}(\alpha, G^0), \quad n = 1, 2, \dots, \infty \quad (3a)$$

$$\theta_t \mid (s_{t-1} = n) \sim G^n, \quad o_t \sim F(\theta_t), \quad \text{for } t = 1, \dots, T, \quad (3b)$$

where G^n is the infinite mixture model for state n , G^0 is the base distribution on the parameters θ , and $F(\theta)$ is the conditional distribution of an observation given the parameter θ (e.g., for continuous observations $F(\theta)$ may represent a Gaussian with mean and covariance defined by θ , while for discrete observations θ may represent the parameters of a multinomial distribution). We also note that after the Markovian state transition from time $t-1$ to time t , we transit from s_{t-1} to the discrete state with which the atom θ_t is associated, and the active infinite mixture model becomes G^{s_t} .

A key component of the iHMM is that the base DP distribution G^0 , from which the atoms in all $\{G^n\}_{n=1}^{\infty}$ are drawn, is itself a draw from a separate DP. This guarantees that the state-dependent observation parameters are identical for all $\{G^n\}_{n=1}^{\infty}$, as required for an HMM. Hence, the discreteness of G^0 (as shown in (2)) is a key property exploited here. To make these properties

explicit, note that (3a) and (3b) imply that:

$$G^0 = \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k^*} \quad G^n = \sum_{k=1}^{\infty} w_k^n \delta_{\theta_k^*} \quad \mathbf{w}^n \sim \text{DP}(\alpha, \beta) \quad n = 1, \dots, \infty, \quad (4)$$

where β and $\mathbf{w}^n$ are both infinite-dimensional vectors of probabilities that sum to one almost surely, and $\theta_k^* \stackrel{iid}{\sim} H$. While HDP construction of G^0 and G^n realize desired properties concerning the consistency of the state-dependent observation models in the iHMM, it also yields a fairly complicated form for the state transition parameters. As discussed further below, the hierarchical form in (4) undermines the ability to perform VB inference. Hence, we will subsequently refine the form of G^0 , to preserve the desired discreteness properties, while imposing a simpler structure that yields an expression similar to (4), albeit with the opportunity to perform fast VB inference.

Combining (3) and (4), the graphical form of an iHMM from a single set of sequential data (single-task learning, or STL) is shown in Fig. 1, with parameters defined as

$$\begin{aligned} \mathbf{o}_t &| s_t, \{\theta_k^*\}_{k=1}^{\infty} \sim F(\theta_{s_t}^*) & \{\theta_k^*\}_{k=1}^{\infty} | H &\sim H \\ s_t &| s_{t-1}, \{\mathbf{w}^n\}_{n=1}^{\infty} \sim \text{Mult}(\mathbf{w}^{s_{t-1}}) \\ \{\mathbf{w}^n\}_{n=1}^{\infty} &| \alpha, \beta \sim \text{DP}(\alpha, \beta) & \beta | \gamma &\sim \text{Stick}(\gamma), \end{aligned}$$

where $\mathbf{w}^n$ corresponds to the n^{th} row of transition matrix A and $F(\theta_k^*)$ is the conditional distribution of the observation given membership in state k . Each observation is represented with a P -dimensional feature vector $\mathbf{o}_t = [o_t^1, \dots, o_t^P]$ and the feature vector is assumed to be generated from a signal model F . If there are multiple tasks, it has been suggested by [7] that one could induce an additional level in the Bayesian hierarchy, letting a master Dirichlet process couple each of the iHMMs. Such a framework borrows information by allowing two tasks to have identical iHMMs instead of allowing tasks to have separate but dependent iHMMs, as will be considered below.

III. MULTIPLE TASKS LEARNING WITH INFINITE HIDDEN MARKOV MODELS

In Section II we described how to construct an iHMM for a single set of sequential data (single-task learning). Now we consider a learning problem in which we have sequential data collected from J different but possibly related tasks, $\mathcal{D} = \{\mathbf{O}_1, \dots, \mathbf{O}_J\}$, where $\mathbf{O}_j = \{o_1^j, \dots, o_{T_j}^j\}$ is the sequential data from task j . For example, each $\mathbf{O}_j$ may represent the observation sequence of

Fig. 1. The infinite hidden Markov model for single task learning. For observation $\mathbf{o}_t$, s_{t-1} defines the infinite mixture model to be used and s_t selects the mixture component according to infinite dimension weight vector $\mathbf{w}^{s_{t-1}}$.

features extracted from the j^{th} music clip, from a set of J music clips. We here assume a single sequence with fixed length T for each task (music piece), but this can easily be generalized to multiple sequences with different observation lengths.

Our goal is to learn accurate iHMMs for each of the tasks as well as to learn the inter-task similarities, the latter used for searching and sorting. For example, one may wish to learn which musical pieces are similar to one another. A naive way to do this is to treat each task separately and learn the individual iHMMs, then apply a post-model-development similarity-measure computation. However since these tasks may be related, the data from one task may potentially be useful to help build the model for other tasks, which can effectively reduce the amount of required data per task to achieve a particular performance (this is of particular value when there are limited data per task). We wish to discover the appropriate sharing structure between tasks and use the sharing information to enhance the learned iHMMs. Meanwhile we wish to exploit the sharing structure to obtain the inter-task relationships directly when performing the model learning, without requiring the extra step of a post-learning similarity computation.

We propose a new hierarchical nonparametric Bayesian model for the aforementioned sequential-data MTL. At the bottom level each task is modeled with an iHMM as described in Sec. II, and at the top level the data in the tasks are shared appropriately by imposing a nested Dirichlet process

(nDP) [9] prior on the base distributions of the iHMMs, yielding the nDP-iHMM. There are several appealing properties of the proposed model: (i) the problem of selecting an appropriate number of HMM states is avoided; (ii) the similarity measurement between tasks is obtained directly; (iii) the state-dependent observation models are shared among related tasks and model learning is improved; and (iv) upon sharing, each task still maintains its own transition matrix, which can be used to distinguish the tasks.

A. The Nested Dirichlet Process

The nested Dirichlet process (nDP) has been proposed [9] to perform intra-task and inter-task clustering simultaneously. Assume there are J related data sets, with each containing $\{x_{ji}\}_{i=1}^{N_j}$ exchangeable observations. The parameter θ_{ji} associated with x_{ji} is assumed to be drawn from a group distribution G_j and all $\{G_j\}_{j=1}^J$ are linked via an nDP(η, γ, H):

$$x_{ji} \mid \theta_{ji} \sim F(\theta_{ji}), \quad \theta_{ji} \mid G_j \sim G_j, \quad i = 1, \dots, N_j$$

$$G_j \sim \sum_{k=1}^{\infty} \pi_k \delta_{G_k^*}, \quad j = 1, \dots, J \quad (5)$$

$$G_k^* = \sum_{l=1}^{\infty} \beta_{kl} \delta_{\theta_{kl}^*}, \quad (6)$$

with $\theta_{kl}^* \sim H$, $\beta_k \sim \text{Stick}(\gamma)$ and $\pi \sim \text{Stick}(\eta)$.

Equation (5) implies that the distribution G_j is a stick-breaking process, in which the atoms are themselves stick-breaking processes drawn from $DP(\gamma, H)$. Since $P(G_j = G_{j'}) = \frac{1}{1+\eta} > 0$, the model induces clustering in the space of distributions. Also, the stick breaking construction of G_k^* ensures that marginally $G_j \sim DP(\gamma, H)$ for every j . In the implementation we will also place Gamma priors on γ and η , to allow the data to inform more strongly about clustering.

It has been shown [9] that the prior correlation between two distributions G_j and $G_{j'}$ is $\text{Cor}(G_j, G_{j'}) = \frac{1}{1+\eta} = P(G_j = G_{j'})$. In addition, the correlation between draws from the process can be calculated from (5) and (6), yielding

$$\text{Cor}(\theta_{ji}, \theta_{j'i'}) = \begin{cases} \frac{1}{1+\gamma} & j = j' \\ \frac{1}{(1+\eta)(1+\gamma)} & j \neq j'. \end{cases}$$

The above indicates that the *a priori* correlation between observations coming from the same group is larger than the correlation between observations coming from different groups, which

is an appealing feature. Therefore, the nDP model simultaneously enables clustering the observations across groups as well as clustering the distributions at the task level. This is different from the HDP, in which only data-level clustering across groups is considered, and task-level clustering does not occur (the task-level clustering in the HDP may be implied indirectly).

B. The nDP-iHMM

Based on the iHMM and the nDP, we propose a new hierarchical model to address the MTL problem for sequential data: the nDP-iHMM. In the nDP-iHMM, each set of sequential data are modeled by an iHMM and those iHMMs are properly shared via an nDP prior. To better understand the MTL setting, reconsider (3a). Each of the J tasks has their own set of DP mixture models $\{G_j^n\}_{n=1}^\infty$ for $j = 1, \dots, J$. To encourage sharing among the J tasks, the collection of base distributions $\{G_1^0, \dots, G_J^0\}$ for the iHMMs are drawn from an nDP, implying the state-dependent observation statistics can be shared across tasks:

$$\pi \sim \text{Stick}(\eta), \quad G_k^* \sim \text{DP}(\gamma, H), \quad G_j^0 \sim \sum_{k=1}^\infty \pi_k \delta_{G_k^*}, \quad G_j^n \sim \text{DP}(\alpha, G_j^0) \quad (7a)$$

$$\theta_t^j \mid (s_{t-1}^j = n) \sim G_j^n, \quad o_t^j \sim F(\theta_t^j), \quad t = 1, \dots, T_j, \quad j = 1, \dots, J, \quad (7b)$$

where the script j denotes the j^{th} task, T_j is the associated sequence length, π is a vector constituting the π_k , and the remaining variables are analogous to the STL case in (3). We note that to address MTL with task-level sharing, (3a) is replaced by (7a), with the remaining (7b) essentially the same as (3b), only repeated J times for the J tasks. A graphical model of the nDP-iHMM is shown in Fig. 2.

Two tasks j and j' share the same observation models (mixture components defined in the base distribution) if $G_j^0 = G_{j'}^0 = G_k^*$ for some k . Note that even though the base distribution G_j^0 and $G_{j'}^0$ are identical, the consequent iHMMs can still be different, because the rows of transition matrices are random draws from a Dirichlet process prior centered on the mixture weights of the base distribution. Since $P(G_j = G_{j'}) = \frac{1}{1+\eta} > 0$, the model induces clustering at the task level, and the DP construction of G_j provides the desired data-level clustering (at the level of states within the HMM). Therefore, the nDP-iHMM simultaneously enables clustering the observations at the data level as well as clustering the distributions at the task level.

The hyperparameters reflect the prior knowledge of how similar the tasks are. As $\eta \rightarrow \infty$,

Fig. 2. The graphical representation of the nDP-iHMM. The base distributions are shared via a nDP and the consequent iHMMs are independently generated given the base distribution.

each base distribution G_j^0 is assigned to a different G_k^* , so that all of the atoms in the base distributions are task-specific. In this case, borrowing of information across the tasks relies entirely on shrinkage occurring under H , which may be sensitive to the specification of H . On the other hand, as $\eta \rightarrow 0$, all the base distributions are identical, $G_j^0 = G^0$, so that a type of pooling occurs and borrowing of information across the tasks is strong. Moreover, as $\gamma \rightarrow 0$, each iHMM degenerates to a single-state HMM and the model reduces to parametric-based clustering.

C. MCMC Inference for nDP-iHMM

The inference for the nDP-iHMM is based on a Gibbs sampler. To simplify the discussion, we consider a single sequence of observations from each task. Let $\mathbf{O}^j = \{o_1^j, \dots, o_{T_j}^j\}$ represent the observation sequence from task j and let indicator variable c_j denote the atom G_k^* for which

$G_j^0 = G_{c_j}^*$. The nDP-iHMM mixture model can be written as follows

$$\begin{aligned}
o_t^j &| c_j, s_t^j, \{\theta_{lk}^*\}_{l,k=1}^\infty \sim F(\theta_{s_t^j c_j}^*) \\
\{\theta_{lk}^*\}_{l,k=1}^\infty &| H \sim H \\
s_t^j &| c_j, s_{t-1}^j, \{\mathbf{w}_{nk}^j\}_{n,k=1}^\infty \sim \text{Mult}(\mathbf{w}_{s_{t-1}^j c_j}^j) \\
\{\mathbf{w}_{nk}^j\}_{n,k=1}^\infty &| c_j, \alpha, \{\beta_k\}_{k=1}^\infty \sim \text{DP}(\alpha, \beta_{c_j}) \\
\{\beta_k\}_{k=1}^\infty &| \gamma \sim \text{Stick}(\gamma) \\
c_j &| \boldsymbol{\pi} \sim \text{Mult}(\boldsymbol{\pi}) \quad \boldsymbol{\pi} | \eta \sim \text{Stick}(\eta).
\end{aligned} \tag{8}$$

We employ a stick-breaking process [23] for the top-level nDP and a Chinese restaurant process [24] for the bottom level iHMM. Similar to the work in [25], we truncate the top level stick-breaking representation to K components. The initialization of the inference includes nDP mixture weights $\{\pi_k\}_{k=1}^K$, center index c_j , hidden state sequences $\{s_t^j\}_{t=1}^T$, iHMM parameters $\{\text{iHMM}_{jk}\}_{k=1}^K$ for $j = 1, \dots, J$ and hyperparameters η, γ and α . The iHMM parameter iHMM_{jk} includes the transition matrix A_j , the observation matrix B_k and the initialization probabilities P_j . We put Gamma priors on the hyperparameters and repeat the following steps until the Gibbs sampler converges:

- 1) Draw new center index c'_j according to $p(c'_j = k) \propto \pi_k \cdot p(\mathbf{O}^j | \text{iHMM}_{jk})$ for every task j .
- 2) For those tasks whose center indices are changed $c'_j \neq c_j$, generate new hidden state sequence s_t^j using the parameter $\text{iHMM}_{jc'_j}$. Recalculate the unique HMM transition matrix A_j for $j = 1, \dots, J$ and the shared HMM observation matrix B_k for $k = 1, \dots, K$. Both of the A 's and B 's are represented as counting matrices.
- 3) Sample new hidden state sequence s_t^j for each iHMM

$$\begin{aligned}
p(s_t^j = l | \mathbf{S}_j^{-t}, \mathbf{O}^j, c_j) &= \\
p(s_t^j = l | s_{t-1}^j = r, s_{t+1}^j = q, o_t^j, c_j = k) &\propto \\
\begin{cases} (n_{jrl}^{-rs_{jt}} + \alpha\beta_{kl}) \frac{\alpha\beta_{kq} + n_{lq}}{\alpha + \sum_{l'=1}^L n_{ll'}} f_l^{-o_t^j}(o_t^j), & \text{if } l \in 1, \dots, L; \\ \alpha\beta_{kl}\beta_{kq} f_{l^{\text{new}}}^{-o_t^j}(o_t^j), & \text{if } l = l^{\text{new}}, \end{cases}
\end{aligned}$$

with $\mathbf{S}_j^{-t}$ being the hidden state sequence excluding s_{jt} , s'_{jt} the previous sampled value of s_{jt} , n_{jrl} the count of transitions from state value r to state value l in task j , and β_{kl} the

mixing weight for state l given the task is using G_k^* .

- 4) Sample new A 's by counting the transitions using s_{jt} 's. Sample new B 's (mixture components) according to its posterior distribution. Generate new iHMM $_{jk}$ parameters from the pair of A_j and B_k .
- 5) Sample nDP mixture weight π_k from $\pi_k = v_k \prod_{i=1}^{k-1} (1 - v_i)$, $v_k \sim \text{Beta}(1 + m_k, \eta + \sum_{s=k+1}^K m_s)$, $k = 1, \dots, K - 1$, $v_K = 1$, where $m_k = \sum_{j=1}^J I(c_j = k)$.
- 6) Update the Gamma priors on the hyperparameters, and sample new η , γ and α .

IV. NESTED DIRICHLET DISTRIBUTION HMM AND VARIATIONAL INFERENCE

The aforementioned Gibbs sampler for the nDP-iHMM involves simple steps for sampling from standard distributions and is relatively simple to implement. However, such an MCMC inference algorithm is computationally prohibitive when considering a large number of tasks. Therefore, we seek to develop a fast variational Bayesian (VB) inference algorithm to learn our MTL model.

Let $\beta_k \sim \text{Stick}(\gamma)$ represent the probability weights on the atoms within G_k^* , and $\mathbf{w}_{nk}^j \sim \text{DP}(\alpha, \beta_k)$ represent the probability weights associated with G_j^n when $G_j = G_k^*$. Unfortunately, within the graphical model of the nDP-iHMM, the terms associated with β_k , the likelihood $p(\mathbf{w}_{nk}^j) \sim \text{DP}(\alpha, \beta_k)$ and the prior $p(\beta_k) \sim \text{Stick}(\gamma)$, are all beta-distributed, and hence there is no conjugacy or analytical update form for β_k . Therefore, while an MCMC inference for (8) is possible, a direct VB analysis is not. To address this problem, one may perform an approach like Monte Carlo EM [26] to sample β_k inside each VB iteration. However, the Monte Carlo EM is very expensive computationally given the need to run a MCMC chain within each VB step.

Recall from the discussion above that each iHMM state has its own multinomial distribution $\text{Mult}(\mathbf{w}^{s_{t-1}})$, where the set of vectors $\{\mathbf{w}^n\}_{n=1}^\infty$ is drawn iid from a Dirichlet process $\text{DP}(\alpha, \beta)$, with β drawn from a stick-breaking process $\text{Stick}(\gamma)$. The transition from state s_{t-1} to s_t occurs with probability $\text{Mult}(\mathbf{w}^{s_{t-1}})$, and the observation o_t is drawn from $F(\theta_{s_t}^*)$, where the set of atoms $\{\theta_k^*\}_{k=1}^\infty$ are drawn iid from the base distribution H . In this formulation we therefore note that the state transition parameters $\{\mathbf{w}^n\}_{n=1}^\infty$ are controlled by two scalar parameters, α and γ , and it is this two-parameter dependency that causes the difficulty with a VB implementation.

In a truncated stick-breaking representation [25] of the iHMM, the iHMM is truncated to L

states, with improving accuracy with increasing L . The finite set of atoms $\{\theta_k^*\}_{k=1}^L$ are again drawn iid from H . In the truncated iHMM each of the weight vectors in the set $\{\mathbf{w}^n\}_{n=1}^L$ represents an L -dimensional probability weight (rather than infinite-dimensional probability weights, as in the untruncated iHMM), drawn as $\mathbf{w}^n \sim \text{DP}(\alpha, \beta)$. As discussed in [23], a draw $\mathbf{w}^n$ from a *Dirichlet distribution*, $\mathbf{w}^n \sim \text{Dir}(\alpha, \beta)$, for L -dimensional base probability weight β , may be constructed as $\mathbf{w}^n = \sum_{k=1}^{\infty} p_k \delta_{Y_k}$, with $p \sim \text{Stick}(\alpha)$ and $Y_k \sim \text{Mult}(\beta)$; this is the same construction as that associated with $\text{DP}(\alpha, \beta)$. Therefore, in the context of the truncated stick-breaking representation of the iHMM, with L sticks/states, the state transition parameters are drawn from a *Dirichlet distribution*. Hence, the draw from $\text{DP}(\alpha, \beta)$, for a finite-dimensional probability weight β , is equivalent to a draw from $\text{Dir}(\alpha, \beta)$. The two-layer form of the *truncated* iHMM implies that the L -dimensional probability weight β , the base of the Dirichlet distribution, is drawn from a truncated version of $\text{Stick}(\gamma)$; it is the dependence of $\{\mathbf{w}^n\}_{n=1}^L$ on α and γ that undermines a VB implementation. We therefore here propose the following formulation change. The probability weights $\{\mathbf{w}^n\}_{n=1}^L$ are now drawn from $\text{Dir}(\alpha; (1/L, 1/L, \dots, 1/L))$, therefore employing a *fixed* base $(1/L, 1/L, \dots, 1/L)$ (rather than a draw from a truncated $\text{Stick}(\gamma)$), and the precision α is again drawn from a Gamma distribution. Through use of this framework, the model preserves the desired conjugacy throughout, permitting variational Bayesian inference. To emphasize that the truncated iHMM yields draws from *Dirichlet distributions* (DDs), we refer to this as a DD-HMM model (rather than the truncated iHMM).

The DD-HMM model may be expressed as (a slight modification of (5))

$$\begin{aligned} \mathbf{o}_t \mid s_t, \{\theta_k^*\}_{k=1}^L &\sim F(\theta_{s_t}^*) & \{\theta_k^*\}_{k=1}^L \mid H &\sim H \\ s_t \mid s_{t-1}, \{\mathbf{w}^n\}_{n=1}^L &\sim \text{Mult}(\mathbf{w}^{s_{t-1}}) \\ \{\mathbf{w}^n\}_{n=1}^L \mid \alpha &\sim \text{Dir}(\alpha, (1/L, 1/L, \dots, 1/L)), \end{aligned} \tag{9}$$

where L sticks/states are considered. Considering (1), this model is equivalent to constituting the $p(o_t | s_{t-1} = n) = \sum_{k=1}^L w_k^n F(o_t | \theta_k^*)$ by drawing the L atoms $\{\theta_k^*\}_{k=1, L}$ iid from H *once* and reusing these L atoms for all n , and then drawing the L -dimensional vectors $\mathbf{w}^n$ via L draws from $\text{Dir}(\alpha, (1/L, 1/L, \dots, 1/L))$ (with the atoms $\{\theta_k^*\}_{k=1, L}$ shared across the different pmfs $\{\mathbf{w}^n\}_{n=1, L}$). We note that in the limit $L \rightarrow \infty$ the probability $p(o_t | s_{t-1} = n)$ so constituted corresponds to a draw from a Dirichlet process $DP(\alpha, H)$.

In the context of multi-task learning, the new formulation is equivalent to a truncated iHMM with the G_k^* in the original nested formulation replaced by $G_k^* = \frac{1}{L} \sum_{l=1}^L \delta_{\theta_{kl}^*}$, with $\theta_{kl}^* \sim H$. We therefore refer to this as a nested DD-HMM, or nDD-HMM for short. Note that in this nDD-HMM, we only alter the bottom iHMM to a DD-HMM for VB implementation, and the top task-level sharing remains intact via an nDP prior. As discussed further below, the nDD-HMM probability of inter-task sharing, driven by the parameter η , is the same as for the nDP-iHMM.

In this nDD-HMM representation we again have the desired discreteness property of the base distribution G_j^0 ($G_j^0 = G_k^*$) to ensure the sharing and consistency of the observation models, but now the transition probabilities w^n are only controlled by variable α , rather than via a two-level structure, which makes direct VB inference intractable. As shown further below, the MCMC nDP-iHMM and the VB nDD-HMM yield comparable results for all test cases considered, except that the VB version scales to large problems significantly better. The excellent performance of the VB-based nDD-HMM relative to the MCMC-based nDP-iHMM suggests the appropriateness of the nDD-HMM; note that these comparisons were done for tens of music pieces, for which the MCMC calculations were feasible. For a large database example with 2500 music pieces, the VB version typically converges in roughly 20 iterations while the MCMC version is infeasible.

A. Variational Bayesian Inference for the nDD-HMM

With the nDD-HMM framework we are able to obtain a fast inference algorithm via the variational Bayesian (VB) method. To consider standard VB inference [13], the top level is truncated to K “sticks” ($\{\pi_k, G_k^*\}_{k=1}^K$), and the stick-breaking DP representation for the base distribution is truncated to L sticks. As discussed above, in the nDD-HMM, the original infinite-dimensional β_k characteristic of the stick-breaking representation of $G_k^* \sim \text{DP}(\gamma, H)$ is now replaced by a L -dimensional β_k corresponding to a uniform probability mass function.

Similar to the nDP-iHMM, the mathematical form of the nDD-HMM mixture model can be

written as

$$\begin{aligned}
o_t^j &| c_j, s_t^j, \{\theta_{lk}^*\}_{l=1 \sim L}^{k=1 \sim K} \sim F(\theta_{s_t^j c_j}^*) \\
\{\theta_{lk}^*\}_{l=1 \sim L}^{k=1 \sim K} &| H \sim H \\
s_t^j &| c_j, s_{t-1}^j, \{\mathbf{w}_{nk}^j\}_{n=1 \sim L}^{k=1 \sim K} \sim \text{Mult}(\mathbf{w}_{s_{t-1}^j c_j}^j) \\
\{\mathbf{w}_{nk}^j\}_{n=1 \sim L}^{k=1 \sim K} &| c_j, \alpha, \{\boldsymbol{\beta}_k\}_{k=1}^K \sim \text{DP}(\alpha, \boldsymbol{\beta}_{c_j}) \\
\boldsymbol{\beta}_k &= [\frac{1}{L}, \dots, \frac{1}{L}] \\
c_j &| \boldsymbol{\pi} \sim \text{Mult}(\boldsymbol{\pi}) \quad \boldsymbol{\pi} | \eta \sim \text{Stick}(\eta).
\end{aligned} \tag{10}$$

The nDD-HMM (10) is almost the same as the nDP-HMM (8), except that both the top and bottom level DPs are represented as truncated stick-breaking processes and $\boldsymbol{\beta}_k$ is set to a constant vector, which is essential to perform the VB inference. For both the nDP-iHMM and the nDD-HMM the task level sharing is controlled by $c_j | \boldsymbol{\pi} \sim \text{Mult}(\boldsymbol{\pi})$ with $\boldsymbol{\pi} | \eta \sim \text{Stick}(\eta)$, and therefore the task level sharing in these models is controlled in exactly the same manner; the only difference in the models is in how the task-dependent HMMs are constituted.

Assume there are N_j observation sequences from task j , and therefore in the standard VB derivation we have: (i) the observed data $Y = \{\mathbf{O}_{jn}\}_{n=1 \sim N_j}^{j=1 \sim J}$; (ii) the hidden variables $X = \{\mathbf{S}_{jn}, c_j\}_{j=1}^J$, with $\mathbf{S}_{jn} = \{s_1^{jn}, \dots, s_T^{jn}\}$ representing the n^{th} hidden state sequence for task j ; and (iii) the parameters $\Theta = \{\boldsymbol{\pi}, \{\boldsymbol{\beta}_k\}_{k=1}^K, \{\{\mathbf{w}_{nk}^j\}_{n=1 \sim L}^{k=1 \sim K}\}_{j=1}^J, \{\theta_{lk}^*\}_{l=1 \sim L}^{k=1 \sim K}\} = \{\boldsymbol{\pi}, \{\boldsymbol{\beta}_k\}, \{\mathbf{B}^k\}, \{\mathbf{P}^{jk}, \mathbf{A}^{jk}\}\}$, where $\mathbf{A}^{jk} = \{\mathbf{w}_{nk}^j\}_{n=1}^L$ is the transition matrix, $\mathbf{P}^{jk} = \mathbf{w}_{0k}^j$ is the initial state probability and $\mathbf{B}^k = \{\theta_{lk}^*\}_{l=1}^L$ is the observation matrix. The set of $\{\boldsymbol{\beta}_k\}$ are fixed at uniform L -dimensional vectors, as indicated above. The triple $\{\mathbf{A}^{jk}, \mathbf{P}^{jk}, \mathbf{B}^k\}$ defines the unique iHMM for modeling task j given $G_0^j = G_k^*$. For simplicity of exposition, we have assumed fixed hyperparameters α and η . However, Gamma priors are placed on the hyperparameters in our experiments.

In VB inference, we wish to maximize a lower bound on the log marginal likelihood $\log p(Y)$ with respect to the $q(X, \Theta)$:

$$\log p(Y) \geq E_q[\log p(X, Y, \Theta)] - E_q[\log q(X, \Theta)],$$

where $q(X, \Theta)$ is the variational distribution and is assumed to be fully factorized:

$$q(X, \Theta) = \prod_{j=1}^J \prod_{n=1}^{N_j} q(\mathbf{S}_{jn}) \prod_{j=1}^J q(c_j) \prod_{k=1}^K \{q(\pi_k)q(\boldsymbol{\beta}_k)q(\mathbf{B}^k)\} \prod_{j=1}^J \prod_{k=1}^K \{q(\mathbf{A}^{jk})q(\mathbf{P}^{jk})\}, \quad (11)$$

where $q(c_j)$ is a multinomial distribution $\text{Mult}(\phi_{j1}, \dots, \phi_{jK})$, $q(\pi_k)$ is a Beta distribution $\text{Be}(\gamma_{k,1}, \gamma_{k,2})$ and $q(\boldsymbol{\beta}_k)$ is a constant vector. We set a truncated stick breaking process as the variational distribution of $\mathbf{P}^{jk}$, and each row of $\mathbf{A}^{jk}$ and $\mathbf{B}^k$. The variational distribution of hidden states, $q(\mathbf{S}_{jn})$, is calculated by the forward-backward algorithm, following the idea in [13]. In the VB algorithm, we iteratively update all terms in (11) to maximize the lower bound, and the detailed VB update equations are shown in the appendix.

V. EXPERIMENTS

We demonstrate the effectiveness of our proposed methods on both synthetic data and real music data. Our synthetic problem, for which the ground truth is known, demonstrates how the nDP-iHMM discovers the underlying sharing structure among multiple sequential data sets. We then apply both MCMC-based nDP-iHMM and VB-based nDD-HMM to a ten music pieces data set to explore the similarities between tasks. Finally, we employ the nDD-HMM to model a large digital music database, for which the nDP-iHMM is infeasible to implement, and show how the learned inter-task similarities can be used in a graph-diffusion analysis to perform sorting and ranking of sequential databases.

A. Synthetic Data

We apply the nDP-iHMM for discovering the relationships between 12 synthetic data sets. Each data set contains 50 sequences of length 20, generated from a distinct discrete HMM. For all HMMs, the number of states $N = 2$ and the codebook size $M = 8$. The parameters of the HMMs have the form of $\{\pi_j = \bar{\pi}_j, \mathbf{A}_j = \bar{\mathbf{A}}_j + \boldsymbol{\epsilon}_j^A, \mathbf{B}_j = \bar{\mathbf{B}}_j + \boldsymbol{\epsilon}_j^B\}$, where $\boldsymbol{\epsilon}_j^A = \begin{bmatrix} \epsilon_{j,11}^A & \epsilon_{j,12}^A \\ \epsilon_{j,21}^A & \epsilon_{j,22}^A \end{bmatrix}$ and $\boldsymbol{\epsilon}_j^B = \begin{bmatrix} \epsilon_{j,11}^B & \epsilon_{j,12}^B & \epsilon_{j,13}^B & \epsilon_{j,14}^B & \epsilon_{j,15}^B & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & \epsilon_{j,24}^B & \epsilon_{j,25}^B & \epsilon_{j,26}^B & \epsilon_{j,27}^B & \epsilon_{j,28}^B \end{bmatrix}$. Each non-zero element in $\boldsymbol{\epsilon}_j^A$ and $\boldsymbol{\epsilon}_j^B$ is independently drawn from a uniform distribution on the interval $[0, 0.05]$. The HMM parameters are chosen as in Table I, therefore there are three clusters among these 12 tasks.

HMM parameters	$\bar{\pi}_j$	A_j	B_j
$j = 1, 2, 3$	0.6950 0.3050	0.8 0.2 0.2 0.8	0.05 0.1 0.7 0.1 0.05 0.0 0.0 0.0 0.0 0.0 0.0 0.05 0.1 0.7 0.1 0.05
$j = 4, 5, 6, 7$	0.8724 0.1276	0.2 0.8 0.8 0.2	0.0 0.0 0.05 0.1 0.7 0.1 0.05 0.0 0.0 0.05 0.1 0.7 0.1 0.05 0.0 0.0
$j = 8, \dots, 12$	0.4729 0.5271	0.5 0.5 0.5 0.5	0.05 0.0 0.0 0.0 0.05 0.1 0.7 0.1 0.1 0.05 0.0 0.0 0.0 0.05 0.1 0.7

TABLE I
HMM PARAMETERS FOR THE SYNTHETIC PROBLEM.

The nDP-iHMM is applied to clustering the tasks. The H is set to be a Dirichlet distribution with parameters all equal to 1, the top level stick-breaking DP is truncated to $K = 20$ components and $Ga(1, 1)$ is imposed on hyperparameters η , γ and α (we considered larger values of K with minimal change in the results). We initialize all tasks with the same center indices and let the nDP-iHMM infer the true underlying relationships between the tasks. The result is shown by the Hinton diagram plotted in Fig. 3(a). In the Hinton diagram, the size of the green box is proportional to the degree of similarity between two tasks. The similarity measure corresponds to the posterior probability that two tasks are grouped together, which can be calculated by the proportion of Gibbs sampling draws where tasks are assigned to the same cluster. Note that this approach relies on soft probabilistic clustering, so that the posterior mean estimates of the base distribution for two tasks always differ, but these estimates eventually converge as the posterior probability of clustering increases. The evolution of center indices is shown in Fig. 3(b). It can be seen that the nDP-iHMM clusters the tasks nicely even when we initialize with the same center indices. Although not shown here for brevity, the nDD-iHMM model, with similar parameter settings to the nDP-iHMM, yields virtually identical sharing as in Fig. 3(a) (for this simple example the VB solution was relatively invariant to the random parameter initialization).

B. Real Music Data

Now we demonstrate the application of our proposed methods to analyzing real data, and in particular, music data. In this experiment, we have ten 1-minute music clips extracted from different pieces. The reason for choosing part of the piece instead of the whole piece is that we are relatively confident on the similarities of those clips. In this way we are able to control the ground

Fig. 3. (a) Hinton diagram of between-task similarities for the synthetic problem. (b) The evolution of center indices versus Gibbs iterations for the 12 tasks.

truth for the real application using our proposed model. These ten clips were chosen deliberately with the following intended clustering: 1) clip 1 is unique in style and instrumentation; 2) clips 2 and 3, 4 and 5, 6 and 7, and 9 and 10 are intended to be paired together; 3) clip 8 is also unique, but is of the same format (instrumentation) as clips 6 and 7 (the names of the pieces are given in Fig. 4).

Our goal is to learn the relationships between the clips, i.e., the similarities of these clips. Meanwhile, we wish to learn an accurate iHMM for each of the clips simultaneously. Each music clip is sampled at 22 kHz and 10-dimensional Melfrequency cepstral coefficient (MFCC) [8] features are extracted for every 25 ms non-overlapping frame. The feature vectors across all the ten clips are concatenated to perform vector quantization (VQ) [27], mapping each feature vector to a code within a VQ codebook of size 32. In our experiment, we choose a sequence of 1 second windows, or 40 observations. Therefore each music clip is transformed into 60 data sequences with 40 observations inside each sequence.

We compare three methods for iHMM model learning: (i) the proposed nDP-iHMM MTL method, (ii) a DP-based MTL method, DP-iHMM, for which a master level DP is used to couple all the iHMMs, and (iii) STL-iHMM – the single task-learning method, for which each clip is analyzed in isolation. Method (i) was discussed in Section IIIB, method (iii) was discussed in Section II, and method (ii) was discussed at the very end of Section II (and was proposed in [7]). Both (ii) and (iii) can be considered as special cases of nDP-iHMM with particular choices

Fig. 4. (a) The average testing sequence likelihood using nDP-iHMM, DP-iHMM and STL-iHMM; (b) the Hinton diagram of between-clip similarities based on sampled center indices in nDP-iHMM for the case of 20 training sequences.

of hyperparameters. The model learning performance is evaluated by calculating the average of testing-sequence likelihood averaged first within clip and then over all the clips. For each clip, a certain number of training sequences are selected and the remaining sequences are used as testing data for that clip-dependent iHMM. The training data are chosen from the middle of the clip, because there may be a quiet period in the two ends. To have a comprehensive comparison, different training set sizes are considered: 2, 4, $\dots$, 10, 20 and 30 sequences are used. All three methods are implemented via Gibbs samplers and the Raftery and Lewis test [28] is performed to determine the number of iterations needed for convergence. All results shown in Fig. 4 are based on 20709 samples obtained after a burn-in period of 2098 iterations. The setting of parameters are the same as in Section V-A. Figure 4(a) shows that the proposed nDP-iHMM method consistently outperforms the other two methods, and the improvement is more dramatic when there is only a small amount of training data available. This is because the nDP-iHMM exploits the inter-task relationships when performing model learning. As a result, the nDP-iHMM directly obtains the similarities between tasks, which are shown in Fig. 4(b). It is clear that the nDP-iHMM captures the between-clip similarities quite well.

While the nDP-iHMM provides a mechanism to obtain the inter-task similarities directly, the STL-iHMM and the DP-iHMM do not. Nevertheless, one may use an appropriate distance measure to compute the similarity based on the learned iHMMs. For this purpose, we use a

Fig. 5. (a) the between-clip similarity matrix computed by (12) using nDP-iHMM results for the case of 6 training sequences; (b) and (c) are same as (a) but using DP-iHMM and STL-iHMM respectively.

distance measure similar to that considered by Aucouturier [29]. The distance between two iHMMs is defined as

$$D(\text{iHMM}_i, \text{iHMM}_j) = \frac{1}{2K} \left\{ \sum_{i=1}^K [\log p(\mathbf{S}_i | \text{iHMM}_i) - \log p(\mathbf{S}_i | \text{iHMM}_j)] + \sum_{j=1}^K [\log p(\mathbf{S}_j | \text{iHMM}_j) - \log p(\mathbf{S}_j | \text{iHMM}_i)] \right\},$$

where $\mathbf{S}_i$'s are sequences simulated from iHMM_i and $\mathbf{S}_j$'s are sequences simulated from iHMM_j .

The similarity between clip i and clip j is then calculated as

$$\text{Sim}(i, j) = \exp\left(-\frac{|D(\text{iHMM}_i, \text{iHMM}_j)|^2}{\sigma^2}\right), \quad (12)$$

where the variance σ^2 is arbitrary. We compute similarities of clips using (12) for the case of 6 training sequences, and plot the Hinton diagrams for all the three methods in Figs. 5(a), (b) and (c), respectively. We observe that the nDP-iHMM (Fig. 5(a)) does the best in discovering the sharing structure of the music clips with limited training data.

While the MCMC-based nDP-iHMM yields pleasing results on MTL for sequential data, the computation is relatively slow. For this ten music pieces example, the nDP-iHMM requires around 40 hours to perform 20000 Gibbs sampling steps on a laptop. The expensive computational cost hinders the nDP-iHMM from solving large-scale problem. On the other hand, the VB inference algorithm converges very fast and learns the ten music clips in about 2 mins on the same machine. Figure 6 shows that the VB-based nDD-HMM trained using 20 sequences per clip

Fig. 6. (a) The evolution of lower bound during VB learning. (b) The inter-clips similarity matrix learned using the VB nDD-HMM.

yields comparable results as the MCMC-based nDP-iHMM, but greatly reduces the computational time, and makes large scale problems solvable. This is further demonstrated below.

C. Large Scale Sequential Databases

Consider the application of the nDD-HMM on a large-scale problem of music analysis. We demonstrate that by treating the multiple musical pieces in a multi-task learning setting via the nDD-HMM, we obtain: (i) a direct similarity measure between musical pieces; and (ii) more accurate individual task-dependent iHMMs, also avoiding the model-selection problem. The similarity matrix learned from (i) is then normalized to define a random walk on a graph, where the nodes on the graph correspond to the different musical pieces; this normalized graph matrix is then applied in a graph-diffusion analysis [14], for music search and sorting.

We examine the algorithm on a music database with 2500 one-minute clips, including classical, jazz and rock music. The classical portion of the database contained works spanning over 200 years and vary from small piano pieces to larger orchestral compositions. The jazz portion was also varied between soft and upbeat works utilizing a variety of instruments. The rock portion spanned primarily the decade 1965-1975 and varied from softer, acoustic works to loud, guitar driven rock. The feature extraction is the same to the one described in Section V-B and the VQ codebook size is 200 ¹.

¹The sequential features and the raw music clips are available to all interested researchers, upon request to the authors.

We conduct two experiments on this large music database. The first experiment is designed for demonstration purposes. We select 151 music clips including 48 pieces from Bach’s *The Well-Tempered Clavier: Book I*, 48 pieces from seven albums by Miles Davis, and 55 pieces from four albums by The Beatles. The reason for choosing these clips instead of the whole 2500 pieces is that we are confident in the similarities of those clips; in this way we are able to provide a reasonable approximation to the ground truth for this real application.

We compare three methods for music similarity measurement: (i) the proposed nDD-HMM; (ii) the STL-iHMM, followed by the distance-measure calculation discussed in (12), and (iii) the nDP-iGMM, in which each task is represented by an infinite Gaussian mixture model [30]. All of the three methods are implemented with the VB algorithm. For the nDD-HMM, we obtain the between-task similarity using the output of variational distribution on $q(c_j)$, which indicates how likely the base distribution of task j uses G_k^* . We select the index $k_j^* = \operatorname{argmax}_k \phi_{jk}$ as the cluster indicator for task j , and then the between-task similarity is simply defined by these cluster indicators. In our experiment, the nDD-HMM uses truncation $K = 20$ and $L = 15$; we also considered a larger number of sticks at both levels, with minimal change in the results. The nDP-iGMM uses the same setting as the nDD-HMM to obtain the similarity matrix.

The similarity-measure performance of the three methods is represented in the colormap, shown in Fig. 7(a)(b)(c). The higher the value in the similarity matrix, the more similar the two clips are. The results are averaged over 20 random VB initializations and all sequences are used for training (this averaging is performed to mitigate or at least minimize effects of local-optimal VB solutions, which are initialization-dependent). We observe that the nDD-HMM result clearly clusters the 151 music clips into three major groups, which conforms with the genre information provided by the ground truth. The STL-iHMM result roughly finds three principal clusters, however there are less relationships considered between clips within the same genre. The nDP-iGMM also discovers the major genre information but with some confusion among the groups; this is because the iGMM considers observations as exchangeable data and does not exploit the sequential information.

To compare the learned task-dependent iHMMs, we did the same test as considered in Fig. 4(a). The results of nDD-HMM and STL-iHMM are shown in Fig. 7(d), and it can be seen that the proposed nDD-HMM method consistently outperforms the STL-iHMM. The performance improvement is more dramatic when there is a small amount of training data available; this is

Fig. 7. Comparison of the between-clip similarity matrix: (a) nDD-HMM results; (b) STL-iHMM results by applying (12) on the learned models; (c) nDP-iGMM results; (d) Average testing sequence likelihood using the iHMMs learned from nDD-HMM and STL-iHMM.

because of the intrinsic sharing mechanism of the nDD-HMM, in which data-level clustering and clip-level clustering are performed simultaneously. The fact that our proposed MTL method learns better iHMMs than the STL counterpart implies that the nDD-HMM can effectively reduce the number of training samples required to achieve a particular performance. Therefore, with the same computational resources, we can obtain similarity information between more music pieces, which is useful for on-demand music recommendation system. We show the example results in Fig. 8, where we use 20 sequences per task to find the inter-clip similarities. For a fair comparison, the similarity matrices are obtained by first learning the iHMMs then applying distance measure (12) for both the nDD-HMM and the STL-iHMM. As we can see, by using only one third of the data, the nDD-HMM already discovers the relationships among the clips, while the STL-iHMM learns the iHMMs independently and does not provide much information about the inter-task similarities.

Fig. 8. Between-clip similarity measurement by first learning iHMMs then applying distance measure (12). 20 sequences per task (1/3 of the total data) are used, and two methods are compared. (a) results for the nDD-HMM; (b) results for the STL-iHMM.

In our second experiment, all 2500 music clips are used and our goal is to obtain the between-task similarity, and to use this within a graphical analysis for music sorting and search. We apply the nDD-HMM to this data set and use the same setting as in the first experiment. Clips 1 - 525 are jazz, 526 - 1525 are rock and 1526 - 2500 are classical music (recall from the discussion above, these data constitute a relatively high level of diversity, that provides a reasonable challenge for the search/sort engine). Sixty observation sequences from each clip are used and the result is shown in Fig. 9(a). The result clearly groups 2500 music clips into three clusters, which correspond to the three major genres. In addition, Fig. 9(a) indicates that there are different within-genre similarities, which will be exploited by the graph-diffusion analysis. The similarity matrix reflected in Fig. 9(a) was appropriately normalized, to define the Markovian properties of a random walk on a graph [14]. The resulting eigenvalue spectrum is shown in Fig. 9(b), and the relatively low dimension of the graph-diffusion space indicates that the manifold is relatively simple, even though a fairly large number of pieces was considered. A graph-diffusion analysis was then performed on this matrix [14], and this was then used to sort the music. The representative search/sort results are reflected in Table II. Although it is difficult to demonstrate without listening, results from these relatively well-known pieces speak to the relatively high-quality sorting results observed. The experiment was implemented with Matlab 7.0 on an Intel Xeon processor with 3 GHz CPU. The required CPU time for one run of the 2500-piece database was approximately 3 hours given the stop threshold of the VB algorithm (that the relative change of the lower bound is smaller than 10^{-3} , with 60 sequences used per piece).

Fig. 9. (a) Between-clip similarity measurement using the nDD-HMM on a large database with 2500 music clips. The three blocks corresponding to jazz, rock and classical music. (b) Eigenvalue spectrum of the normalized graph-diffusion matrix (calculated using the result from (a)).

Artist Search	Bach (Classical)	John Coltrane (Jazz)	The Beatles (Rock)
Composition Search	WTC Book I, Prelude No 2	Tommy's Tune	Hello Goodbye
Top ten most similar Playlist	Bach - WTC Book I, Prelude No 16 Bach - Sinfonia No 11 Bach - WTC Book II, Prelude No 16 Beethoven - Bagatelle Op 126 No 2 Brahms - Handel Variations No 10 Brahms - Piano Quintet Op 34 Mvt IV Bach - Toccata BWV 915 Schumann - Piano Sonata No 2 Mvt I Schumann - Piano Sonata No 2 Mvt III Bach - WTC Book I, Prelude No 11	John Coltrane - Anatomy Charlie Parker - Cosmic Rays John Coltrane - One by One Charlie Parker - My Heart Belongs to Daddy John Coltrane - Love Thy Neighbor John Coltrane - I Hear a Rhapsody Miles Davis - Someday My Prince Will Come John Coltrane - Big Paul John Coltrane - Cattin' Charlie Parker - Now's the Time	The Beatles - Getting Better The Who - Drowned The Who - Pure and Easy The Beatles - Carry That Weight The Who - Cut My Hair Led Zeppelin - Down By the Seaside The Who - 1921 The Who - Is it in my Head The Rolling Stones - Shine A Light The Who - Rael 1

TABLE II
EXAMPLE MUSIC SEARCH-ENGINE RESULTS, BASED ON NDD-HMM AND GRAPH DIFFUSION.

VI. CONCLUSION

We have proposed a new hierarchical Bayesian model for multi-task learning with sequential data. The model aims to simultaneously perform task-level clustering as well as data-level clustering. Each task is modeled by an infinite hidden Markov model (iHMM), solving the fundamental model selection problem in HMM learning, and all the iHMMs are shared appropriately via a nested Dirichlet process (nDP). A MCMC inference engine has been implemented for the nDP-iHMM, and promising results are obtained on a synthetic problem and a real music data set.

To analyze large-scale sequential databases, the nDP-iHMM is truncated to an nDD-HMM, to allow a variational Bayesian inference algorithm. Encouraging nDD-HMM results have been demonstrated on a large-scale database involving 2500 musical clips. The framework effectively learns better models than the competing methods, and the learned inter-task (inter-music-piece)

relationship is used to define the Markovian properties of a random walk on a graph, on which graph-diffusion-based data sorting is applied. Therefore, the nDD-HMM can be used for modeling and sorting large sequential databases, such as the application of the music recommendation system.

APPENDIX

We show the detailed VB update equations for the variational distribution $q(X, \Theta)$ described in (11). The derivations are based on VB-DP [31] and VB-HMM [13], with modifications to accommodate the nDD-HMM model.

- $q(\pi_k), k = 1, \dots, K$. To consider a stick-breaking representation, we write $\pi_k = \pi'_k \prod_{t=1}^{k-1} (1 - \pi'_t)$ and set $q(\pi'_k) \sim \text{Be}(\gamma_{k,1}, \gamma_{k,2})$. We update the variational distribution on $q(\pi'_k)$ as

$$\gamma_{k,1} = 1 + \sum_{j=1}^J \phi_{jk}, \quad \gamma_{k,2} = \eta + \sum_{j=1}^J \sum_{t=k+1}^K \phi_{jt}, \quad q(\pi'_K) = \delta(\pi'_K - 1).$$

where $k = 1, \dots, K - 1$ and $\phi_{jk} = p(c_j = k)$.

- $q(\mathbf{P}^{jk}, \mathbf{A}^{jk}, \mathbf{B}^k), k = 1, \dots, K, j = 1, \dots, J$. We put stick-breaking prior as the variational distribution on each row of the matrices:

$$\begin{aligned} q(p_l^{jk'}) &\sim \text{Be}(r_{l,1}^{P^{jk}}, r_{l,2}^{P^{jk}}), \quad p_l^{jk'} = p_l^{jk'} \prod_{t=1}^{l-1} (1 - p_t^{jk'}) \\ q(a_{nl}^{jk'}) &\sim \text{Be}(r_{nl,1}^{A^{jk}}, r_{nl,2}^{A^{jk}}), \quad a_{nl}^{jk'} = a_{nl}^{jk'} \prod_{t=1}^{l-1} (1 - a_{nt}^{jk'}) \\ q(b_{nm}^{k'}) &\sim \text{Be}(r_{nm,1}^{B^k}, r_{nm,2}^{B^k}), \quad b_{nm}^{k'} = b_{nm}^{k'} \prod_{t=1}^{m-1} (1 - b_{nt}^{k'}), \end{aligned}$$

and the update equations are

$$\begin{aligned} r_{l,1}^{P^{jk}} &= 1 + \sum_{k=1}^K \phi_{jk} w_l^{P^{jk}}, \quad r_{l,2}^{P^{jk}} = \alpha + \sum_{k=1}^K \sum_{t=l+1}^L \phi_{jk} w_t^{P^{jk}} \quad p_L^{jk'} = 1 \\ r_{nl,1}^{A^{jk}} &= 1 + \sum_{k=1}^K \phi_{jk} w_{nl}^{A^{jk}}, \quad r_{nl,2}^{A^{jk}} = \alpha + \sum_{k=1}^K \sum_{t=l+1}^L \phi_{jk} w_{nt}^{A^{jk}} \quad a_{nL}^{jk'} = 1 \\ r_{nm,1}^{B^k} &= 1 + \sum_{j=1}^J \phi_{jk} w_{nm}^{B^k}, \quad r_{nm,2}^{B^k} = u_B + \sum_{j=1}^J \sum_{t=m+1}^M \phi_{jk} w_{nt}^{B^k} \quad b_{nM}^{k'} = 1 \\ n &= 1, \dots, L, \quad l = 1, \dots, L - 1, \quad m = 1, \dots, M - 1. \end{aligned}$$

where M is the size of codebook in iHMMs.

- $q(\mathbf{S}_{jn}), j = 1, \dots, J, n = 1, \dots, N_j$. Similar to the VB-HMM [13], we calculate the ex-

pectation of $\mathbf{S}_{jn}$ using forward-backward algorithm with data $\mathbf{O}_{jn}$ and modified parameters

$$\begin{aligned}\tilde{p}_l^{jk} &= \exp[(\Psi(r_{l,1}^{Pjk}) - \Psi(r_{l,1}^{Pjk} + r_{l,2}^{Pjk})) + \sum_{t=1}^{l-1} (\Psi(r_{t,2}^{Pjk}) - \Psi(r_{t,1}^{Pjk} + r_{t,2}^{Pjk}))] \\ \tilde{a}_{nl}^{jk} &= \exp[(\Psi(r_{nl,1}^{Ajk}) - \Psi(r_{nl,1}^{Ajk} + r_{nl,2}^{Ajk})) + \sum_{t=1}^{l-1} (\Psi(r_{nt,2}^{Ajk}) - \Psi(r_{nt,1}^{Ajk} + r_{nt,2}^{Ajk}))] \\ \tilde{b}_{nm}^k &= \exp[(\Psi(r_{nm,1}^{Bk}) - \Psi(r_{nm,1}^{Bk} + r_{nm,2}^{Bk})) + \sum_{t=1}^{m-1} (\Psi(r_{nt,2}^{Bk}) - \Psi(r_{nt,1}^{Bk} + r_{nt,2}^{Bk}))].\end{aligned}$$

The corresponding outputs are $\{w_l^{Pjk}\}_{l=1}^L$, $\{w_{nl}^{Ajk}\}_{n,l=1}^L$ and $\{w_{nm}^{Bk}\}_{n=1 \sim L}^M$.

- $q(c_j)$, $j = 1, \dots, J$. We define $\phi_{jk} = p(c_j = k) \propto \exp(\rho_{j,k})$ with

$$\begin{aligned}\rho_{j,k} &= \sum_{l=1}^L w_l^{Pjk} \ln(\tilde{p}_l^{jk}) + \sum_{n=1}^L \sum_{l=1}^L w_{nl}^{Ajk} \ln(\tilde{a}_{nl}^{jk}) + \sum_{n=1}^L \sum_{m=1}^M w_{nm}^{Bk} \ln(\tilde{b}_{nm}^k) \\ &\quad + \mathbf{1}(k < K) [\Psi(\gamma_{k,1}) - \Psi(\gamma_{k,1} + \gamma_{k,2})] \\ &\quad + \mathbf{1}(k > 1) \sum_{t=1}^{k-1} [\Psi(\gamma_{t,2}) - \Psi(\gamma_{t,1} + \gamma_{t,2})] \quad k = 1, \dots, K.\end{aligned}$$

REFERENCES

- [1] R. Caruana, “Multitask learning,” *Machine Learning*, vol. 28, pp. 41–75, 1997.
- [2] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum, “Hierarchical topic models and the nested Chinese restaurant process,” in *Neural Information Processing Systems*, 2004.
- [3] S. Thrun and J. O’Sullivan, “Discovering structure in multiple learning tasks: The TC algorithm,” in *The 13th International Conference on Machine Learning*, 1996.
- [4] B. Bakker and T. Heskes, “Task clustering and gating for Bayesian multitask learning,” *Journal of Machine Learning Research*, vol. 4, pp. 83–99, 2003.
- [5] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram, “Multi-task learning for classification with Dirichlet process priors,” *Journal of Machine Learning Research*, vol. Jan., pp. 35–63, 2007.
- [6] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, Eds., *Bayesian data analysis*. Chapman and Hall, 1995.
- [7] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, “Hierarchical Dirichlet processes,” *Journal of the American Statistical Association*, vol. 101(476), pp. 1566–1581, 2006.
- [8] B. Logan and A. Salomon, “A music similarity function based on signal analysis,” in *IEEE International Conference on Multimedia and Expo*, 2001.
- [9] A. Rodriguez, D. B. Dunson, and A. E. Gelfand, “The nested Dirichlet process (with discussion),” *Journal of the American Statistical Association*, vol. to appear, 2007.
- [10] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, pp. 257–286, 1989.

- [11] P. Runkle, P. K. Bharadwaj, L. Couchman, and L. Carin, “Hidden Markov models for multi-aspect target classification,” *IEEE Transactions on Signal Processing*, vol. 47, pp. 2035–2040, 1999.
- [12] A. Stolcke and S. Omohundro, “Hidden Markov model induction by Bayesian model merging,” in *Advances in Neural Information Processing Systems*, vol. 5, 1993.
- [13] M. J. Beal, “Variational algorithms for approximate Bayesian inference,” Ph.D. dissertation, Gatsby Computational Neuroscience Unit, University College London, 2003.
- [14] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis, “Diffusion Maps, Spectral Clustering and Eigenfunctions of Fokker-Planck Operators,” in *Advances in Neural Information Processing Systems*, vol. 18, 2005.
- [15] R. Ando and T. Zhang, “A framework for learning predictive structures from multiple tasks and unlabeled data,” *Journal of Machine Learning Research*, vol. 6, pp. 1817–1853, 2005.
- [16] S. Ben-David and R. Schuller, “Exploiting task relatedness for multiple task learning,” in *Proceedings of Computational Learning Theory (COLT)*, 2003.
- [17] T. Evgeniou and M. Pontil, “Regularized multi-task learning,” in *Proceedings ACM Special Interest Group on Knowledge, Discovery and Data Mining*, 2004.
- [18] T. Jebara, “Multi-task feature and kernel selection for svms,” in *Proceedings Int. Conference Machine Learning (COLT)*, 2004.
- [19] A. Argyriou, T. Evgeniou, and M. Pontil, “Multi-task feature learning,” in *Advances in Neural Information Processing Systems 19*, 2007.
- [20] M. J. Beal, Z. Ghahramani, and C. Rasmussen, “The infinite Hidden markov model,” in *Neural Information Processing Systems*, 2002.
- [21] M. D. Escobar and M. West, “Bayesian density estimation and inference using mixtures,” *Journal of the American Statistical Association*, vol. 90, pp. 577–588, 1995.
- [22] T. S. Ferguson, “A Bayesian analysis of some nonparametric problems,” *Annals of Statistics*, vol. 1, no. 2, pp. 209–230, 1973.
- [23] J. Sethuraman, “A constructive definition of Dirichlet priors,” *Statistica Sinica*, vol. 4, pp. 639–650, 1994.
- [24] D. Aldous, “Exchangeability and related topics,” in *Ecole d’Ete St Flour 1983*, 1985, pp. 1–198.
- [25] H. Ishwaran and L. F. James, “Gibbs sampling methods for stick-breaking priors,” *Journal of the American Statistical Association*, vol. 96, pp. 161–173, 2001.
- [26] W. Jank, “Implementing and diagnosing the stochastic approximation EM algorithm,” *Journal of Computational and Graphical Statistics*, vol. 15, pp. 803–829, 2006.
- [27] Y. Linde, A. Buzo, and R. M. Gray, “An algorithm for vector quantizer design,” *IEEE Trans. Communications*, vol. COM-28, pp. 84–95, 1980.
- [28] A. E. Raftery and S. M. Lewis, “How many iterations in the Gibbs sampler?” *Bayesian Statistics*, vol. 4, pp. 763–773, 1992.
- [29] J. J. Aucouturier and F. Pachet, “Music similarity measures: What’s the use?” in *International Symposium on Music Information Retrieval (ISMIR)*, 2002.
- [30] C. Rasmussen, “The infinite Gaussian mixture model,” in *Neural Information Processing Systems*, 2000.
- [31] D. Blei and M. Jordan, “Variational Inference for Dirichlet process mixtures,” *Journal of Bayesian Analysis*, vol. 1(1), pp. 121–144, 2005.