Generating structure of latent variable models for nested data

Masakazu Ishihata
NTT Communication Science Laboratories
ishihata.masakazu@lab.ntt.co.jp

Tomoharu Iwata
NTT Communication Science Laboratories
iwata.tomoharu@lab.ntt.co.jp

Abstract

Probabilistic latent variable models have been successfully used to capture intrinsic characteristics of various data. However, it is nontrivial to design appropriate models for given data because it requires both machine learning and domain-specific knowledge. In this paper, we focus on data with nested structure and propose a method to automatically generate a latent variable model for the given nested data, with the proposed method, the model structure is adjustable by its structural parameters. Our model can represent a wide class of hierarchical and sequential latent variable models including mixture models, latent Dirichlet allocation, hidden Markov models and their combinations in multiple layers of the hierarchy. Even when deeply-nested data are given, where designing a proper model is difficult even for experts, our method generate an appropriate model by extracting the essential information. We present an efficient variational inference method for our model based on dynamic programming on the given data structure. We experimentally show that our method generates correct models from artificial datasets and demonstrate that models generated by our method can extract hidden structures of blog and news article datasets.

1 INTRODUCTION

Probabilistic latent variable models have been successfully used for analyzing and capturing intrinsic characteristics of a wide variety of datasets. They are applied to various tasks such as dimension reduction, clustering, visualization and cluster matching [14, 28, 16, 15]. However, designing appropriate models for given data is difficult because it requires machine learning knowledge to formulate models and derive algorithms, and also domain-specific knowledge to introduce appropriate latent variables and their dependencies.

In this paper, we focus on data with a nested structure and aim to automatically generate a latent variable model that is appropriate to the given nested data. Many data have nested structures such as hierarchies. A document contains sentences and each of the sentences contains words. Purchase data consist of purchase histories of many users, where a user history contains multiple shopping events and a shopping event is represented as a basket containing items that are purchased at the same time. Also music shows such hierarchy; many musicians compose music scores by combining multiple phrases, where each of the phrases consists of musical notes. Such nested data sometimes contain not only hierarchical information but also sequential information, that is, the order of elements in nested groups such as word order in a sentence. Even though such rich structural information is given, we do not need to incorporate all of the information in models. For example, a latent Dirichlet allocation (LDA) model [7], which is a widely-used Bayesian latent variable model for text data, assumes a document is a bag of words; it ignores word order. In contrast, a hidden Markov model (HMM), which is the most famous latent variable model for sequential data, assumes a document is a sequence of words but ignores its hierarchical information. When we design models for such structured data, we have to extract essential information and reflect that into models by introducing several modeling assumptions.

The main contribution of this paper is threefold. First, we propose a latent variable model based on a hierarchical and sequential structure of the given data, where our model can adjust its model structure by *structural parameters*. By changing the values of structural parameters, it can represent various models including mixture models, LDA models, HMMs and their combinations. Second, we propose a universal variational inference algorithm for our model based on dynamic programming on the given data structure. Even if model structures and data structures are changed, we can efficiently compute variational free energy

(VFE), which is used for a model scoring function [5], by our universal algorithm. Third, we formulate the model generation task as an optimization problem for maximizing VFE with respect to structural parameters. No matter how deep the input nested data are, our method extracts the essential information by adjusting model structure. Our method can generate complex models if necessary. Otherwise, it generates simple models.

The remainder of this paper is organized as follows. We first briefly review related work in Section 2. We then introduce an ordered tree representation for nested data in Section 3. In Section 4 we propose our latent variable model and our model generation method. We describe its efficient variational inference algorithm in Section 5. In Section 6, we illustrate our method by experiments using artificial and real-world datasets. Finally, we summarize this paper in Section 7.

2 RELATED WORK

A number of methods for automatically generating model strictures from data have been proposed. Structure learning for Bayesian networks (BNs) [22] is one such example. A BN defines a joint distribution over random variables by a directed acyclic graph (DAG) and model parameters. A DAG defines conditional independencies over random variables and model parameters define their conditional distributions. BN structure learning (BNSL) [17] is a problem to find the DAG that maximizes a scoring function from a model class. Unfortunately, the problem is generally NP-hard [8, 9]. Most existing BNSL methods assume no latent variable or limit the number of latent variables [5, 24, 26, 27, 20]. Whereas our aim is to obtain appropriate latent variable models for the given data, the aim of BNSL is to obtain the exact dependencies of observable variables. It is intractable to naively apply existing BNSL methods to our aim because introducing latent variables exponentially increases the number of candidate model structures and the computation time of a scoring function. Our method avoids such intractability by restricting our model class based on the given nested structure and by introducing a universal algorithm for efficient score computation on the restricted model class.

A hierarchical latent class (HLC) model [28] is a tree-structured BN with latent variables, where its leaf nodes are observable variables and the others are latent variables. Learning HLC model structure is similar to our aim, however, it differs in the following two points. First, our method assumes that data have nested structure and exploits the data structure to generate appropriate models. Second, our method considers both hierarchical and sequential dependencies of latent variables. Model structure generated by our method includes, but are not limited to, tree structures.

A model generation method based on matrix factorization has been proposed [12]. The method defines its model class by using context-free grammar of which grammatical rules correspond to matrix factorization processes. A sentence generated by the grammar represents a hierarchical matrix factorization model. The method assumes that given data are represented as a single matrix whereas our method uses data together with its nested structure.

Several latent variable models for nested data have been proposed. The segmented topic model (STM) [11], which is a variant of probabilistic topic models [6], assumes that a document is a collection of segments and each of the segments is a collection of words. The STM captures correlations of topics over segments and segments over documents by introducing segment-level and document-level topic probabilities. The tree-informed LDA (tiLDA) [18], which is a multi-level LDA model, exploits another type of hierarchical structure of documents. The tiLDA model assumes that documents are grouped into categories and each of the categories are also grouped into another categories. Those models exploit inner-document and outer-document nested structures, respectively. Our model also copes with both inner and outer document hierarchies. It additionally exploits sequential information such as the word and segment order. By automatically extracting the essential part of such hierarchical and sequential structures, it generates appropriate models for the given nested data.

Similarly, latent variable models for sequential data have been proposed. A hidden Markov topic model (HMTM) [3], which is a natural extension of LDA models for the nonexchangeable setting, assumes that word topics are generated by a Markov chain. Each document has its own topic Markov chain, however, word emission probabilities of topics are shared by all documents. A hidden topic Markov model (HTMM) [13] introduces a Markov chain to LDA models with a different fashion from HMTM. The HTMM assumes that words in the same sentence share a common topic and each of the sentence topics is generated by a Markov chain. Similar to those methods, our method can combine LDA models and HMMs. Additionally, our method automatically chooses appropriate levels in which Markov chains should be introduced.

3 ORDERED TREE FOR NESTED DATA

We here introduce an ordered tree representation of nested data. Suppose that nested data $\boldsymbol{D}$ is represented as a pair $(\boldsymbol{x}, \boldsymbol{T})$, where $\boldsymbol{x} \equiv (x_n)_{n=1}^{N}$ is a sequence of observable variables and $\boldsymbol{T}$ is an ordered tree representing its nested structure. An ordered tree $\boldsymbol{T}$ is defined by a triplet $(\mathrm{N}, \mathit{par}, \mathit{sib})$, where $\mathrm{N} = \{0, \ldots, N\}$ is node indexes in the breadth first order (i.e., 0 is a root node), and $\mathit{par} : \mathrm{N} \to \mathrm{N}$ and $\mathit{sib} : \mathrm{N} \to \mathrm{N}$ define the parent and the previous sibling of each node, respectively. Thus, $\mathit{par}(n)$ denotes the n's

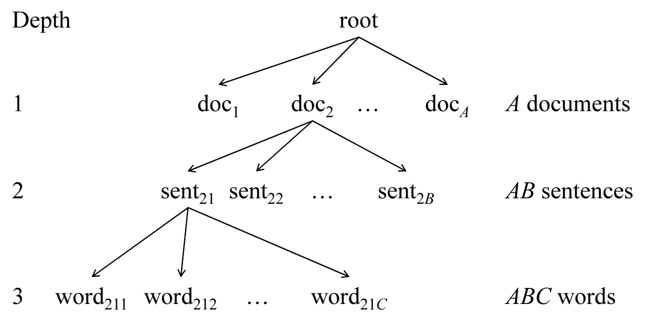
Depth
root
1
doc$_1$ doc$_2$ … doc$_A$
A documents
2
sent$_{21}$ sent$_{22}$ … sent$_{2B}$
AB sentences
3
word$_{211}$ word$_{212}$ … word$_{21C}$
ABC words

Figure 1: An ordered tree representation of a 3-layered nested structure.

parent and $sib(n)$ denotes the n's previous sibling, where $sib(n)=0$ denotes that n has no previous sibling. Let D be the depth of $\boldsymbol{T}$ and d_n be that of n. Also let N_d $(1 \le d \le D)$ be a set of nodes n such that $d_n = d$. An observable variable x_n is associated with each node n, where variables in the same depth are assumed to share the same domain. For simplicity, we focus on the case that x_n $(n \in \mathrm{N}_d)$ has a discrete domain $\{1, \ldots, V_d\}$, although our method can be extended easily to the continuous case, where $V_d = 0$ denotes that $n \in \mathrm{N}_d$ has no observation.

We illustrate the ordered tree representation of a collection of documents with a 3-layered nested structure in Figure 1. Suppose that the collection contains A documents, where each document contains B sentences and each sentence consists of C words. A node in the first layer $n \in \mathrm{N}_1$ corresponds to a document, and $n \in \mathrm{N}_2$ and $n \in \mathrm{N}_3$ correspond to a sentence and a word, respectively. The tree depth is $D=3$ and the tree size is $N = A + AB + ABC$. Each leaf node $n \in \mathrm{N}_3$ has a corresponding observation x_n which is a term in V_3-term vocabulary. Each inter node $n \in \mathrm{N} \backslash \mathrm{N}_3$ have no observation that is denoted by $V_1 = V_2 = 0$.

When we design a latent variable model for the above nested data, we choose essential structural information and reflect it in the model. In LDA models, for example, only document-level information is preserved and the other information (the sentence and word order) is ignored. In this paper, we aim to automatically extract important information in $\boldsymbol{T}$ and generate an appropriate model for given nested data $\boldsymbol{D}$.

4 OUR MODEL

In this section, we first describe our latent variable model $\boldsymbol{M}$ of which dependencies among latent variables are adjustable by its *structural parameters*. We then propose a local search method to optimize the parameters for given nested data $\boldsymbol{D}$. The method generates an appropriate model structure of $\boldsymbol{M}$ that captures intrinsic characteristics of $\boldsymbol{D}$.

Table 1: An assumption variable A_d and dependencies.

A_d	Explanation	Dependency
I-det	Index-deterministic	$z_n = n$
P-det	Parent-deterministic	$z_n = z_\ell$
N-dep	Non-dependent	$z_n \perp\!\!\!\perp z_\ell$, $z_n \perp\!\!\!\perp z_m$
P-dep	Parent-dependent	$z_n \not\perp\!\!\!\perp z_\ell$, $z_n \perp\!\!\!\perp z_m$
S-dep	Sibling-dependent	$z_n \perp\!\!\!\perp z_\ell$, $z_n \not\perp\!\!\!\perp z_m$
B-dep	Both-dependent	$z_n \not\perp\!\!\!\perp z_\ell$, $z_n \not\perp\!\!\!\perp z_m$

4.1 MODEL DEFINITION

We here describe our model $\boldsymbol{M}$ which is a Bayesian latent variable model for nested data $\boldsymbol{D} = (\boldsymbol{x}, \boldsymbol{T})$. Our model $\boldsymbol{M}$ is defined by a quadruplet $(\boldsymbol{T}, \boldsymbol{A}, \boldsymbol{\alpha}, \boldsymbol{\beta})$, where $\boldsymbol{T}$ is the given ordered tree, $\boldsymbol{A}$ is *structural parameters* which control dependencies among latent variables, and $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are model parameters.

We first introduce latent variables $\boldsymbol{z} \equiv (z_n)_{n=1}^N$ and structural parameters $\boldsymbol{A} \equiv (A_d)_{d=1}^D$, where A_d is an *assumption variable* which defines dependencies among latent variables in the dth layer. Our model $\boldsymbol{M}$ assumes that each node $n \in \mathrm{N}_d$ has a discrete latent variable $z_n \in \{1, \ldots, K_d\}$. We define $z_0 = 0$ for denoting the root 0 has no latent variable. Each observable variable x_n is generated by depending only on the corresponding latent variable z_n. Each latent variable z_n is generated by depending on latent variables of n's parent and sibling $z_{par(n)}$ and $z_{sib(n)}$. For simplicity, let $\ell = par(n)$ and $m = sib(n)$ in this paper. An assumption variable A_d with a discrete domain $\{\text{I-det}, \text{P-det}, \text{N-dep}, \text{P-dep}, \text{S-dep}, \text{B-dep}\}$ controls the dependency of z_n $(n \in \mathrm{N}_d)$ as shown in Table 1. I-det and P-det denote that z_n deterministically takes a value n and z_ℓ, respectively. N-dep denotes that z_n is independent of other latent variables. P-dep, S-dep and B-dep denote that z_n depends on z_ℓ, z_m and the both, respectively. The structural parameters $\boldsymbol{A}$ control the entire model structure of $\boldsymbol{M}$ which represents dependencies among all latent variables $\boldsymbol{z}$.

We next introduce model parameters $\boldsymbol{\alpha} \equiv (\boldsymbol{\alpha}_d)_{d=1}^D$ and $\boldsymbol{\beta} \equiv (\boldsymbol{\beta}_d)_{d=1}^D$, where $\boldsymbol{\alpha}_d \equiv (\alpha_{d,k})_{k=1}^{K_d}$ and $\boldsymbol{\beta}_d \equiv (\beta_{d,v})_{v=1}^{V_d}$ are parameters of Dirichlet distributions for generating $\boldsymbol{\theta}_{d,i,j} \equiv (\theta_{d,i,j,k})_{k=1}^{K_d}$ and $\boldsymbol{\phi}_{d,k} \equiv (\phi_{d,k,v})_{v=1}^{V_d}$, respectively. Here, $\boldsymbol{\theta}_{d,i,j}$ is a parameter of a categorical distribution for generating z_n $(n \in \mathrm{N}_d)$ and $\boldsymbol{\phi}_{d,k}$ is that for generating x_n, that is, $\boldsymbol{\theta}_{d,i,j,k}$ is a probability of $z_n = k$ given $z_\ell = i$ and $z_m = j$ and $\boldsymbol{\phi}_{d,k,v}$ is a probability of $x_n = v$ given $z_n = k$. The generation process of $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\phi})$ given model $\boldsymbol{M} \equiv (\boldsymbol{T}, \boldsymbol{A}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ is as follow.

1. For each depth $d = 1, \ldots, D$ and hidden class $i = 1, \ldots, K_{d-1}$ and $j, k = 1, \ldots, K_d$
   (a) Draw $\boldsymbol{\theta}_{d,i,j} \sim \mathrm{Dir}(\boldsymbol{\alpha}_d)$
   (b) Draw $\boldsymbol{\phi}_{d,k} \sim \mathrm{Dir}(\boldsymbol{\beta}_d)$

2. For each depth $d=1,\dots,D$ and node $n\in\mathrm{N}_d$
   (a) Choose a class z_n by

$$\begin{array}{ll} \textbf{case } A_d & \\ \textbf{when } \text{I-det}: & z_n := n \\ \textbf{when } \text{P-det}: & z_n := z_\ell \\ \textbf{when } \text{N-dep}: & z_n \sim \mathrm{Cat}(\boldsymbol{\theta}_{d,0,0}) \\ \textbf{when } \text{P-dep}: & z_n \sim \mathrm{Cat}(\boldsymbol{\theta}_{d,z_\ell,0}) \\ \textbf{when } \text{S-dep}: & z_n \sim \mathrm{Cat}(\boldsymbol{\theta}_{d,0,z_m}) \\ \textbf{when } \text{B-dep}: & z_n \sim \mathrm{Cat}(\boldsymbol{\theta}_{d,z_\ell,z_m}) \end{array}$$

   (b) Draw an observation $x_n \sim \mathrm{Cat}(\boldsymbol{\phi}_{d,z_n})$

Here, $\boldsymbol{\theta}_{d,i,j}$ is a parameter for generating z_n ($n\in\mathrm{N}_d$) given $z_\ell=i$ and $z_m=j$, however, we set $i=0$ and $j=0$ when z_n is independent of z_ℓ and z_m, respectively. The joint probability $p(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\theta},\boldsymbol{\phi}\mid\boldsymbol{M})$ factorizes into

$$p(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\theta},\boldsymbol{\phi}|\boldsymbol{M}) = p(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\phi})\,p(\boldsymbol{z}|\boldsymbol{A},\boldsymbol{\theta})\,p(\boldsymbol{\theta}|\boldsymbol{\alpha})\,p(\boldsymbol{\phi}|\boldsymbol{\beta}),$$

and each probability is computed as

$$p(\boldsymbol{\theta}\mid\boldsymbol{\alpha}) = \prod_{d=1}^{D}\prod_{i=0}^{K_{d-1}}\prod_{j=0}^{K_d}\mathrm{Dir}(\boldsymbol{\theta}_{d,i,j};\boldsymbol{\alpha}_d), \quad (1)$$

$$p(\boldsymbol{\phi}\mid\boldsymbol{\beta}) = \prod_{d=1}^{D}\prod_{k=1}^{K}\mathrm{Dir}(\boldsymbol{\phi}_{d,k};\boldsymbol{\beta}_d), \quad (2)$$

$$p(\boldsymbol{z}\mid\boldsymbol{A},\boldsymbol{\theta}) = \prod_{d=1}^{D}\prod_{i=0}^{K_{d-1}}\prod_{j=0}^{K_d}\prod_{k=1}^{K_d}\theta_{d,i,j,k}^{c_{d,i,j,k}(\boldsymbol{A},\boldsymbol{z})}, \quad (3)$$

$$p(\boldsymbol{x}\mid\boldsymbol{z},\boldsymbol{\phi}) = \prod_{d=1}^{D}\prod_{k=1}^{K_d}\prod_{v=1}^{V_d}\phi_{d,k,v}^{c_{d,k,v}(\boldsymbol{z},\boldsymbol{x})}, \quad (4)$$

where $c_{d,i,j,k}(\boldsymbol{A},\boldsymbol{z})$ and $c_{d,k,v}(\boldsymbol{z},\boldsymbol{x})$ are define as

$$c_{d,i,j,k}(\boldsymbol{A},\boldsymbol{z}) \equiv \begin{cases} |\{n\in\mathrm{N}_d \mid z_n=k\}| & A_d=\text{N-dep}, i=j=0 \\ |\{n\in\mathrm{N}_d \mid z_\ell=i, z_n=k\}| & A_d=\text{P-dep}, j=0 \\ |\{n\in\mathrm{N}_d \mid z_m=j, z_n=k\}| & A_d=\text{S-dep}, i=0 \\ |\{n\in\mathrm{N}_d \mid z_\ell=i, z_m=j, z_n=k\}| & A_d=\text{B-dep} \\ 0 & \text{otherwise}, \end{cases}$$

$$c_{d,k,v}(\boldsymbol{z},\boldsymbol{x}) \equiv |\{n\in\mathrm{N}_d \mid z_n=k, x_n=v\}|.$$

Here, $c_{d,k,v}(\boldsymbol{z},\boldsymbol{x})$ is the count of $\phi_{d,k,v}$ used in the generating process and $c_{d,i,j,k}(\boldsymbol{A},\boldsymbol{z})$ is that of $\theta_{d,i,j,k}$.

Our model $\boldsymbol{M}$ includes a variety of well-known existing models. Given text data $\boldsymbol{x}$ and ordered tree $\boldsymbol{T}$ shown in Figure 1 as its nested structure, we can represent various models by adjusting structural parameters $\boldsymbol{A}$ as shown in Table 2, where the corresponding plate notations are shown in Figure 2.

Table 2: Example models for 3-layered text data. MMM denotes a multinomial mixture model. dX, sX and wX denote document-level X, sentence-level X and word-level X, respectively.

	$\boldsymbol{A}=(A_1,A_2,A_3)$	Corresponding Model
(1)	N-dep, P-det, P-det	dMMM
(2)	I-det, P-det, P-dep	dLDA
(3)	I-det, I-det, S-dep	wHMM
(4)	I-det, S-dep, P-dep	sHMM + wMMM
(5)	I-det, P-det, B-dep	dLDA + wHMM
(6)	I-det, B-dep, P-dep	dLDA+sHMM+wMMM

4.2 MODEL GENERATION

Given nested data $\boldsymbol{D}=(\boldsymbol{x},\boldsymbol{T})$, we aim to generate the "best" model $\boldsymbol{M}=(\boldsymbol{T},\boldsymbol{A},\boldsymbol{\alpha},\boldsymbol{\beta})$ for $\boldsymbol{D}$ by optimizing $\boldsymbol{A}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. To define the "best", a criterion for comparing multiple models is needed. A log marginal likelihood $\mathcal{L}[\boldsymbol{M}]\equiv\ln p(\boldsymbol{x}\mid\boldsymbol{M})$ is one of such criteria which measures how model $\boldsymbol{M}$ fits to the given data $\boldsymbol{x}$ [19, 10]. However, computing $\mathcal{L}[\boldsymbol{M}]$ is intractable because it requires an intractable summation $\sum_{\boldsymbol{z}} p(\boldsymbol{x},\boldsymbol{z}\mid\boldsymbol{M})$. We instead employ the variational free energy (VFE) [5] $\mathcal{F}[\boldsymbol{M}]$, which is a lower bound of $\mathcal{L}[\boldsymbol{M}]$. The definition and an efficient computation algorithm of $\mathcal{F}[\boldsymbol{M}]$ are described in Section 5. The algorithm is also used for optimizing model parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ given $\boldsymbol{D}$ and $\boldsymbol{A}$. Even though VFE $\mathcal{F}[\boldsymbol{M}]$ is computable, it is still intractable to maximize $\mathcal{F}[\boldsymbol{M}]$ with respect to $\boldsymbol{A}$ because it requires evaluation of $\mathcal{F}[\boldsymbol{M}]$ for all possible instantiations for $\boldsymbol{A}$. We avoid such intractable evaluation by employing the following local search:

1. Let $\boldsymbol{A}$ s.t. $A_d=$P-det for all d and evaluate $\mathcal{F}[\boldsymbol{M}]$
2. Let $t:=0$ and $\mathrm{B}_t:=\{\boldsymbol{A}\}$
3. Let $\mathrm{NB}_t:=$ all neighbors of $\boldsymbol{A}\in\mathrm{B}_t$
4. Evaluate $\mathcal{F}[\boldsymbol{M}]$ for all $\boldsymbol{A}\in\mathrm{NB}_t$
5. Let $\mathrm{B}_{t+1}:=$ the w best structures w.r.t. VFE
6. Terminate if $\mathrm{B}_{t+1}=\mathrm{B}_t$ and return 3. otherwise

We initialize the first candidate $\boldsymbol{A}$ as the simplest structure which contains only deterministic latent variables that is equivalent to no latent variable model. We then repeat generating neighbors of current candidates and choosing the w best structures as new candidates while VFE increases, where w is the search bandwidth. Here, $\boldsymbol{A}'$ is a neighbor of $\boldsymbol{A}$ if there exists exactly one d such that $A_d\neq A'_d$.

The above method automatically generates an appropriate latent variable model $\boldsymbol{M}$ for the given nested data $\boldsymbol{D}$ by optimizing its structural parameters $\boldsymbol{A}$ and its model parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. It can also automatically select an appropriate K_d, which is the number of clusters in the d-th layer, by running with different K_d and choosing $\boldsymbol{M}$ with the highest VFE. In the case we have a condition on models, our method can generate only expected models by skip-

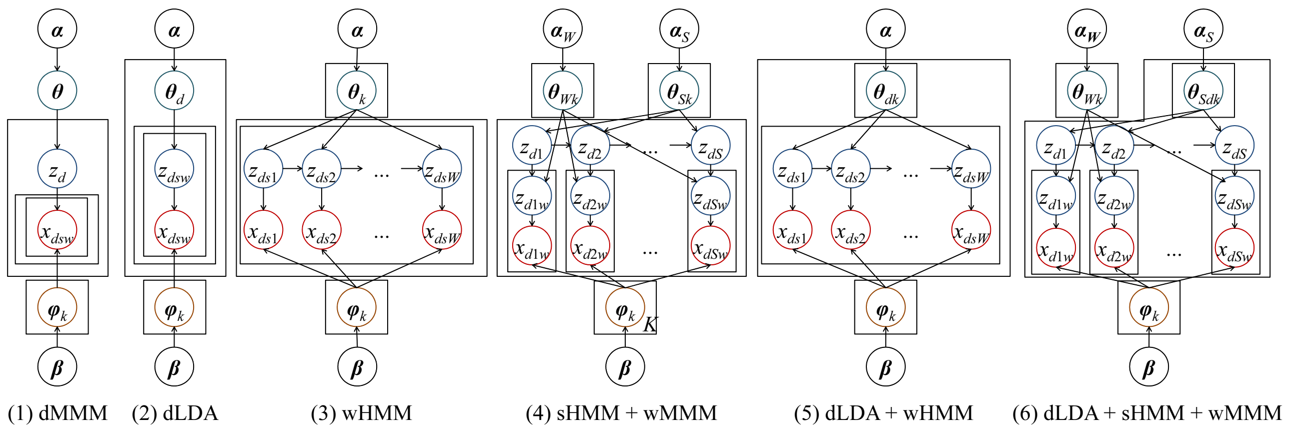

Figure 2: Plate representations of models in Table 2.

ping violated models in the search. For example, a model for the document clustering task should have a latent variable which indicates the cluster index of each document. Our method can generate a such model by excepting models which have no latent variable in the document-layer.

5 VARIATIONAL INFERENCE

We here define a variational free energy $\mathcal{F}[\boldsymbol{M}]$ which is used as a measure how model $\boldsymbol{M}$ fits to given nested data $\boldsymbol{D}$. We propose an efficient method for computing $\mathcal{F}[\boldsymbol{M}]$ that employs a dynamic programming algorithm on $\boldsymbol{T}$ as its building block. The method is universal because it does not require change even if model structure $\boldsymbol{A}$ and data structure $\boldsymbol{T}$ are changed.

5.1 VARIATIONAL FREE ENERGY

Using Jensen's inequality, we obtain the following lower bound of log marginal likelihood $\mathcal{L}[\boldsymbol{M}]$,

$$\begin{aligned}\ln p(\boldsymbol{x} \mid \boldsymbol{M}) &\geq \mathrm{E}_q[\ln p(\boldsymbol{x} \mid \boldsymbol{z}, \boldsymbol{\phi})] + \mathrm{E}_q[\ln p(\boldsymbol{z} \mid \boldsymbol{A}, \boldsymbol{\theta})] \\ &\quad + \mathrm{E}_q[\ln p(\boldsymbol{\theta} \mid \boldsymbol{\alpha})] + \mathrm{E}_q[\ln p(\boldsymbol{\phi} \mid \boldsymbol{\beta})] - \mathrm{H}[q] \\ &\equiv \mathcal{F}[q, \boldsymbol{M}] ,\end{aligned}$$

where q is a variational distribution such that $q(\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\phi}) = q(\boldsymbol{z})\, q(\boldsymbol{\theta})\, q(\boldsymbol{\phi})$ and $\mathrm{H}[q]$ is its entropy. By using the Euler-Lagrange equation, we obtain

$$q(\boldsymbol{z}) \propto \exp\big(\mathrm{E}_{q(\phi)}[\ln p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\phi})] + \mathrm{E}_{q(\theta)}[\ln p(\boldsymbol{z}|\boldsymbol{\theta})]\big) , \quad (5)$$

$$q(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta} \mid \boldsymbol{\alpha}) \exp\big(\mathrm{E}_{q(\boldsymbol{z})}[\ln p(\boldsymbol{z} \mid \boldsymbol{A}, \boldsymbol{\theta})]\big) , \quad (6)$$

$$q(\boldsymbol{\phi}) \propto p(\phi \mid \boldsymbol{\beta}) \exp\big(\mathrm{E}_{q(\boldsymbol{z})}[\ln p(\boldsymbol{x} \mid \boldsymbol{z}, \boldsymbol{\phi})]\big) . \quad (7)$$

By iteratively updating q, we can maximize $\mathcal{F}[q, \boldsymbol{M}]$ w.r.t. q. We can also maximize it w.r.t. model parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by fixed point iteration [21]. We use $\mathcal{F}[\boldsymbol{M}]$ to denote $\mathcal{F}[q, \boldsymbol{M}]$ which is computed by estimated q, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$.

We next detail these q updates. Because $q(\boldsymbol{\theta})$ and $q(\boldsymbol{\phi})$ are approximations of (1) and (2), we define

$$q(\boldsymbol{\theta}) \equiv \prod_{d=1}^{D} \prod_{i=0}^{K_d} \prod_{j=0}^{K_d} \mathrm{Dir}(\boldsymbol{\theta}_{d,i,j}; \boldsymbol{a}_{d,i,j}) ,$$

$$q(\boldsymbol{\phi}) \equiv \prod_{d=1}^{D} \prod_{k=1}^{K} \mathrm{Dir}(\boldsymbol{\phi}_{d,k}; \boldsymbol{b}_{d,k}) ,$$

where $\boldsymbol{a}_{d,i,j} \equiv (a_{d,i,j,k})_{k=1}^{K_d}$ and $\boldsymbol{b}_{d,k} \equiv (b_{d,k,v})_{v=1}^{V_d}$ are variational parameters. By substituting (1)-(4) into (6)(7), their updates are obtained by

$$a_{d,i,j,k} = \alpha_{d,k} + \mathrm{E}_{q(\boldsymbol{z})}[c_{d,i,j,k}(\boldsymbol{A}, \boldsymbol{z})] , \quad (8)$$

$$b_{d,k,v} = \beta_{d,v} + \mathrm{E}_{q(\boldsymbol{z})}[c_{d,k,v}(\boldsymbol{z}, \boldsymbol{x})] . \quad (9)$$

We also obtain the following updates of $q(\boldsymbol{z})$:

$$q(\boldsymbol{z}) \propto \left(\prod_{d=1}^{D} \prod_{i=0}^{K_{d-1}} \prod_{j=0}^{K_d} \prod_{k=1}^{K_d} {\theta^*_{d,i,j,k}}^{c_{d,i,j,k}(\boldsymbol{A},\boldsymbol{z})} \right) \times \left(\prod_{d=1}^{D} \prod_{k=1}^{K_d} \prod_{v=1}^{V_d} {\phi^*_{d,k,v}}^{c_{d,k,v}(\boldsymbol{z},\boldsymbol{x})} \right) ,$$

where

$$\theta^*_{d,i,j,k} \equiv \exp\left(\Psi(a_{d,i,j,k}) - \Psi\left(\sum_{k'=1}^{K_d} a_{d,i,j,k'} \right) \right) ,$$

$$\phi^*_{d,k,v} \equiv \exp\left(\Psi(b_{d,k,v}) - \Psi\left(\sum_{v'=1}^{V_d} b_{d,k,v'} \right) \right) ,$$

and $\Psi(\cdot)$ is a digamma function. Here, $q(\boldsymbol{z}) = p(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$[1] holds by the definitions of (3) and (4).

[1]Even $\boldsymbol{\phi}^*_{d,k}$ and $\boldsymbol{\theta}^*_{d,i,j}$ are not a probability vector, $p(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ is a probability through the effect of a normalizing factor $p(\boldsymbol{x} \mid \boldsymbol{\theta}^*, \boldsymbol{\phi}^*) = \sum_{\boldsymbol{z}} p(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$.

Thus, expectations in (8) and (9) are computed as

$$\mathrm{E}_{q(\boldsymbol{z})}[c_{d,i,j,k}(\boldsymbol{A},\boldsymbol{z})] = \sum_{n\in \mathrm{N}_d} P^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta}^*,\boldsymbol{\phi}^*] \tag{10}$$

$$\mathrm{E}_{q(\boldsymbol{z})}[c_{d,k,v}(\boldsymbol{z},\boldsymbol{x})] = \sum_{n\in \mathrm{N}_d : x_n = v} \sum_{i=0}^{K_{d-1}} \sum_{j=0}^{K_d} P^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta}^*,\boldsymbol{\phi}^*] \tag{11}$$

$$P^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}] \equiv p(z_\ell = i, z_m = j, z_n = k \mid \boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}) .$$

We propose an efficient dynamic programming method for computing $P^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]$ in Section 5.2.

5.2 EFFICIENT PROBABILITY COMPUTATION

For efficient computation of $P^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]$, we define

$$R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}] \equiv p(z_\ell = i, z_m = j, z_n = k, \boldsymbol{x} \mid \boldsymbol{\theta},\boldsymbol{\phi}) . \tag{12}$$

Using the above, $P^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]$ is computed as

$$P^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}] = \frac{R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]}{p(\boldsymbol{x} \mid \boldsymbol{\theta},\boldsymbol{\phi})}, \tag{13}$$

$$p(\boldsymbol{x} \mid \boldsymbol{\theta},\boldsymbol{\phi}) = \sum_{i=0}^{K_{d-1}} \sum_{j=0}^{K_d} R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}] . \tag{14}$$

Computing $R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]$ by naively summarizing out $z_{n'}$ $(n' \in \mathrm{N} \backslash \{\ell, m, n\})$ requires exponential time in N.

We here describe an efficient dynamic programming algorithm for computing $R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]$. Let $\mathrm{Dec}(n)$ be all descendants of n in $\boldsymbol{T}$. Also let $\mathrm{Sib}^-(n)$ and $\mathrm{Sib}^+(n)$ be all younger and older siblings of n, respectively. We then introduce the following four sets, *inside set* $\mathrm{I}(n)$, *outside set* $\mathrm{O}(n)$, *forward set* $\mathrm{F}(n)$ and *backward set* $\mathrm{B}(n)$:

$$\mathrm{I}(n) \equiv \{n\} \cup \mathrm{Dec}(n) , \quad \mathrm{F}(n) \equiv \bigcup_{m\in\{n\}\cup \mathrm{Sib}^-(n)} \mathrm{I}(m) ,$$

$$\mathrm{O}(n) \equiv \mathrm{N}\backslash \mathrm{Dec}(n) , \quad \mathrm{B}(n) \equiv \bigcup_{m'\in\{n\}\cup \mathrm{Sib}^+(n)} \mathrm{I}(m') .$$

Figure 3 shows examples of those sets of a 3-layered ordered tree. Using those sets, a set of all nodes N factorizes into $\mathrm{N} = \mathrm{O}(\ell) \cup \mathrm{F}(m) \cup \mathrm{B}(n)$. For a set $C \subseteq \mathrm{N}$, we define $\boldsymbol{x}_C \equiv (x_n)_{n\in C}$ and $\boldsymbol{z}_C \equiv (z_n)_{n\in C}$. Then, $R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]$ factorizes into

$$\begin{aligned} R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}] = {} & p\big(\boldsymbol{x}_{\mathrm{O}(\ell)}, z_\ell = i \mid \boldsymbol{\theta},\boldsymbol{\phi}\big) \\ & p\big(\boldsymbol{x}_{\mathrm{F}(m)}, z_m = j \mid z_\ell = i, \boldsymbol{\theta},\boldsymbol{\phi}\big) \\ & p\big(\boldsymbol{x}_{\mathrm{B}(n)}, z_n = k \mid z_\ell = i, z_m = j, \boldsymbol{\theta},\boldsymbol{\phi}\big) . \end{aligned}$$

To compute the above, we introduce the following four probabilities, *inside probability* $I_n[k]$, *outside probability* $O_n[k]$, *forward probability* $F_n[i,k]$ and *backward probability* $B_n[i,j,k]$:

$$\begin{aligned} I_n[k] &\equiv p\big(x_{\mathrm{I}(n)} \mid z_n = k, \boldsymbol{\theta},\boldsymbol{\phi}\big) , \\ O_n[k] &\equiv p\big(x_{\mathrm{O}(n)}, z_n = k \mid \boldsymbol{\theta},\boldsymbol{\phi}\big) , \\ F_n[i,k] &\equiv p\big(x_{\mathrm{F}(n)}, z_n = k \mid z_\ell = i, \boldsymbol{\theta},\boldsymbol{\phi}\big) , \\ B_n[i,j,k] &\equiv p\big(x_{\mathrm{B}(n)}, z_n = k, \mid z_\ell = i, z_m = j, \boldsymbol{\theta},\boldsymbol{\phi}\big) . \end{aligned}$$

The above probabilities can be computed in the following dynamic programming manner:

$$I_n[k] = \phi_{d,k,v} B_c[k,0] , \tag{15}$$

$$O_n[k] = \sum_{i=0}^{K_{d-1}} \sum_{j=0}^{K_d} O_n[i,j,k] , \tag{16}$$

$$F_n[i,k] = I_n[k] \sum_{j=1}^{K_d} F_m[i,j]\, \theta_{d,i,j,k}, \tag{17}$$

$$B_n[i,j,k] = I_n[k]\, B_{m'}[i,k]\, \theta_{d,i,j,k}, \tag{18}$$

where $v = x_n$, $d = d_n$, c is the oldest child of n, m' is the next sibling of n and

$$O_n[i,j,k] \equiv O_\ell[k]\, F_m[i,j]\, B_{m'}[i,j]\, \phi_{d,k,v} \theta_{d,i,j,k},$$

$$B_n[i,j] \equiv \sum_{k=1}^{K_d} B_n[i,j,k] .$$

Finally, the target probability is computed by

$$R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}] = O_\ell[i]\, F_m[i,j]\, B_n[i,j,k] . \tag{19}$$

In summary, the variational free energy $\mathcal{F}[\boldsymbol{M}]$ is computed by Algorithm 1. When $K_d = K$ for all d, the complexity of this algorithm is $O(NK^3)$, however, it decreases according to structural parameters $\boldsymbol{A}$. For example, it becomes $O(NK^2)$ if $\boldsymbol{A}$ represents an HMM, and also becomes $O(NK)$ if $\boldsymbol{A}$ represents an LDA model. Note that the naive computation for $R^{i,j,k}_{\ell,m,n}[\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\phi}]$ requires exponential time in N but our method is polynomial.

6 EMPIRICAL RESULTS

6.1 ARTIFICIAL DATASETS

We here illustrate that our method is feasible for generating structures of latent variable models by using artificial datasets, where correct models which generate the datasets are known. We designed 12 models with the 3-layered nested structure $\boldsymbol{T}$ shown in Figure 1 and generated 12 datasets from these models. Each dataset contains L documents, each document contains L sentences and each sentence contains L words. We set the cluster size of each depth as $K_1 = K_2 = K_3 = 5$ and the vocabulary size as $V_1 = V_2 = 0$ and $V_3 = 500$.

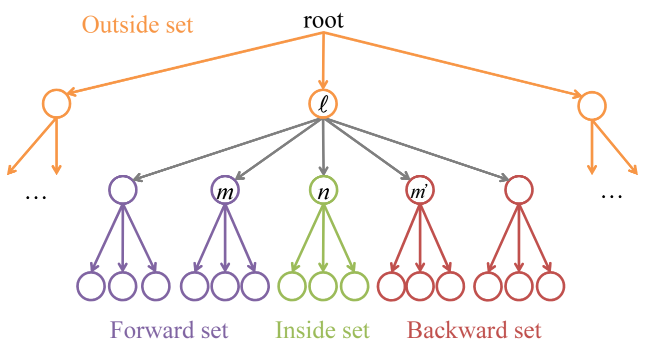

Figure 3: Inside set $\mathrm{I}(n)$, outside set $\mathrm{O}(\ell)$, forward set $\mathrm{F}(m)$ and backward sets $\mathrm{B}(m')$ of a 3-layered ordered tree, where m' is the next sibling of n.

Algorithm 1 Variational Free Energy $\mathcal{F}[\boldsymbol{M}]$

```
repeat
  for $n = 1, \ldots, N$ do
    Compute $I_n[k]$ for each $k$ by (15)
    Compute $B_n[i, j, k]$ for each $i, j, k$ by (18)
  end for
  for $n = N, \ldots, 1$ do
    Compute $F_n[i, k]$ for each $i, k$ by (17)
    Compute $O_n[k]$ for each $k$ by (16)
  end for
  Compute $R^{i,j,k}_{\ell,m,n}[\boldsymbol{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*]$ for each $i, j, k$ by (19)
  Compute $P^{i,j,k}_{\ell,m,n}[\boldsymbol{x}, \boldsymbol{\theta}^*, \boldsymbol{\phi}^*]$ for each $i, j, k$ by (13)
  Compute $\mathrm{E}_{q(\boldsymbol{z})}[c_{d,i,j,k}(\boldsymbol{z})]$ for each $d, i, j, k$ by (10)
  Compute $\mathrm{E}_{q(\boldsymbol{z})}[c_{d,k,v}(\boldsymbol{z})]$ for each $d, k, v$ by (11)
  Update $a_{d,i,j,k}$ for each $d, i, j, k$ by (8)
  Update $b_{d,k,v}$ for each $d, k, v$ by (9)
  Update $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by fixed point iteration [21]
until $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ converge
Return $\mathcal{F}[q, \boldsymbol{M}]$ computed by estimated $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{\alpha}, \boldsymbol{\beta}$
```

We applied our method for the 12 datasets with search bandwidth $w = 3$. Table 3 shows the correct models and generated models. Our method generated the correct models for all dataset with $L = 50$. Even the dataset size is small, it correctly estimated simple models such as MMMs.

6.2 REAL-WORLD DATASETS

To demonstrate its applicability to real data, we applied our method for two real-world datasets, a Japanese blog dataset and an English news dataset. In both experiments, we ran our method with different $K = 10, 20, 30$ and chose the model with the highest VFE. We set search bandwidth $w = 3$ and excluded models of which complexity is $O(NK^3)$ for tractability. We also applied LDA models for the same datasets with different $K = 10, 20, \ldots, 100$ and compared VFEs of LDA models and those of generated models.

Table 4: VFEs for the Japanese blog dataset.

Model	VFE
user-LDA	-5.239×10^6
article-LDA	-5.284×10^6
Generated model	-5.222×10^6

Table 6: VFEs for the English news dataset.

Model	First dataset	Second dataset
day-LDA	-8.739×10^6	-4.891×10^6
article-LDA	-8.299×10^6	-4.609×10^6
sentence-LDA	-8.554×10^6	-4.842×10^6
Generated model	-7.658×10^6	-4.555×10^6

JAPANESE BLOG DATASET. The dataset is a collection of 100 users' goo blog [1] articles from April 7th to June 27th in 2007, where each article is extracted as a sequence of nouns and verbs. We filtered out the 100 most frequent terms as stop words and used the top 5,000 terms to construct a 3-layered nested dataset consisted of 100 users, 7,687 articles and 731,065 words.

For this dataset, our method generated the same model as (5) in Figure 2, which is equivalent to an HMTM [3] that is a combination of a user-level LDA and word-level HMM. The model is better than LDA models from the aspect of VFE as shown in Table 4. In the generated model, each user has a corresponding topic transition matrix but topics are shared by all users. Table 5 shows the ten most frequent topics and their five most probable words, where words were translated from Japanese to English. We manually gave a topic name for each topic. The table shows that the generated model captured latent topics in the Japanese blog dataset.

ENGLISH NEWS DATASET. The dataset is a collection of Reuters' news articles in 1987 [2]. We extracted articles from March 1st to 31st and constructed a 4-layered nested dataset consisting of 29 days, 10,535 articles, 79,155 sentences and 31,057 terms in the vocabulary, where words including 0-9 were replaced to "NUM". We then created two datasets by extracting the 5,000 most frequent terms: the first dataset that we did not filter out any words but the second one that we filtered the first 100 terms as stop words. Those datasets contained 1,369,888 and 626,316 words, respectively.

Figure 4 shows plate representations of models generated by our method. Our method generated a word-level HMM for the first dataset and a 3-layered MMM for the second one. Table 6 shows VFEs of the generated models and LDA models. As shown, our method found better models than LDA in terms of VFE.

Figure 5 shows the ten most frequent topics, their five most probable words and the five most probable topic transi-

Table 3: Correct models and generated models with $L = 10, 30, 50$. Incorrect assumptions are colored in red.

	Correct Model	$L=10$	$L=30$	$L=50$
1.	dMMM	N-dep, P-det, P-det	N-dep, P-det, P-det	N-dep, P-det, P-det
2.	sMMM	I-det, N-dep, P-det	I-det, N-dep, P-det	I-det, N-dep, P-det
3.	dLDA	I-det, P-det, P-dep	I-det, P-det, P-dep	I-det, P-det, P-dep
4.	sLDA	I-det, I-det, P-det	I-det, S-dep, P-det	I-det, I-det, P-dep
5.	dHMM	I-det, P-det, P-dep	S-dep, P-det, P-det	S-dep, P-det, P-det
6.	sHMM	I-det, S-dep, P-det	I-det, S-dep, P-det	I-det, S-dep, P-det
7.	wHMM	P-det, P-det, P-det	N-dep, P-det, B-dep	N-dep, P-det, B-dep
8.	dHMM + wMMM	I-det, P-det, P-dep	I-det, P-det, P-dep	S-dep, P-det, P-dep
9.	sHMM + wMMM	P-det, P-det, P-det	N-dep, B-dep, P-dep	I-det, S-dep, P-dep
10.	dLDA + sHMM	P-det, S-dep, P-det	S-dep, B-dep, P-det	I-det, P-det, B-dep
11.	dLDA + wHMM	P-det, P-det, P-det	S-dep, P-det, B-dep	I-det, B-dep, P-det
12.	sLDA + wHMM + wMMM	P-det, P-det, P-det	S-dep, B-dep, P-det	I-det, B-dep, P-dep

Table 5: The ten most frequent topics and their five most probable words obtained from the Japanese blog data.

z_{uaw}	Topic name	The five most probable words
24	Verbs 1	sleep, go, try, mind, no
5	Stock market	stock, management, company, proportion, market
16	News	book, ads, news, US, company
9	Research	psychology, perception, research, knowledge, description
28	Animation	version, animation, appearance, track, purchase
27	Travel (urban)	bus, station, Rahmen, father, taste
12	Travel (nature)	blossom, weather, Japanese cherry, park, wind
25	Software	case, screen, data, process, layout
19	International news	dictionary, multimedia, phraseology, terrorism, high concept
8	Verbs 2	receive, book, visit, try, time

tions over those topics. The generated model captured frequently-appearing grammatical patterns. A topic transition ($18 \rightarrow 7$) describes a pattern (number $\rightarrow$ unit), and ($\{29, 13\} \rightarrow 26 \rightarrow \{17, 22\}$) describes a pattern (Preposition (1) & (2) $\rightarrow$ Article $\rightarrow$ {Objects, Adjective}). Because such grammatical patterns commonly appear in every sentences, our method extracted the word-layer and removed the other layers.

In contrast, our method generated a 3-layered MMM (day-article-sentence) for the second dataset. This is because grammatical patterns in the dataset were destroyed by filtering out stop words. The model performed day, article and sentence clustering; days were grouped into two clusters and articles and sentences were grouped into 30 clusters. Table 7 shows the three most probable article and sentence clusters and the five most probable words. The day clusters shared neither article cluster nor sentence cluster in the table. Sentence clusters in day cluster 3 describe periodic financial reports, budgets of IT companies and shareholder meeting, respectively. Those in day cluster 26 describe policies of executives, domestic economy and economic relations between Japan and US, respectively. It seems that day cluster 3 tends to mention about periodical events.

7 CONCLUSION

We proposed a method for generating latent variable models from nested data. We presented a latent variable model of which structure is adjustable by its structural parameters. Our model attempts to optimize the parameters by maximizing the variational free energy. We derived a universal variational inference method for our model based on dynamic programming on the given data structure. No matter how deep the input nested data are, our method extracts its essential information and generates a model with an appropriate complexity. We empirically showed that our method correctly generated model structures by synthetic datasets and also showed that it extracted hidden structures of blog and news datasets.

We note a potential direction for future work. We can extend our model to non-parametric Bayesian setting by replacing Dirichlet distributions by HDPs [23]. To the best of our knowledge, it seems difficult to provide an efficient variational inference method for the extended model because it subsumes the infinite hidden Markov model (iHMM) [4] which is a non-parametric Bayesian extension of HMM. However, we can easily derive a Gibbs sampling in the same way as that of iHMM. The beam sampling [25] is an efficient sampling method for iHMM, which com-

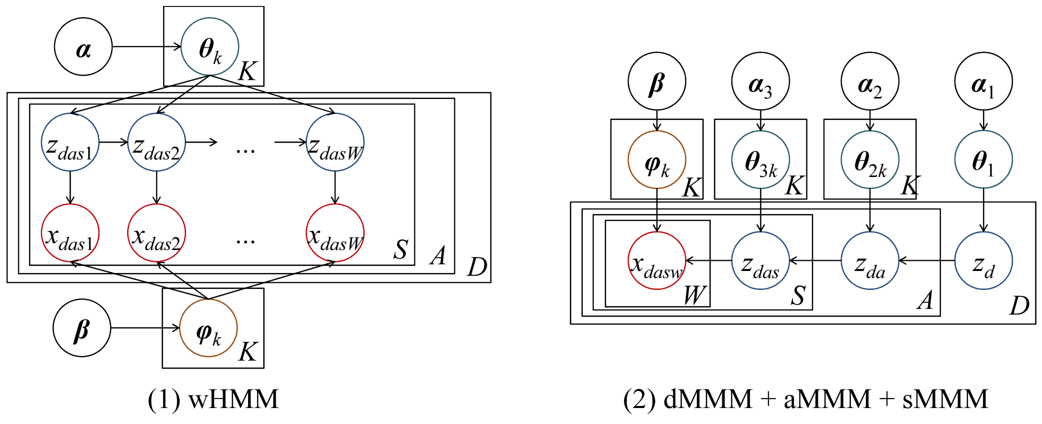

Figure 4: Plate representations of generated models from two English News datasets.

z_{dasw}	Topic Name	The five most probable words
26	Articles	the, a, its, an, this
29	Prepositions (1)	in, to, for, on, by
22	Objects	company, year, bank, week, government
18	Numbers	NUM, two, one, five, three
17	Adjective	new, us, first, common, united
14	Verb (1)	will, is, would, has, was
13	Prepositions (2)	of, in, for, and, from
4	Economic words	oil, foreign, trade, exchange, economic
7	Units	mln, pct, billion, cts, dlrs
19	Verb (2)	said, that, told, added, but

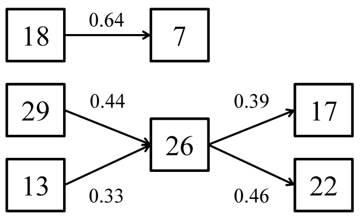
$$\begin{tikzcd}
\boxed{18} \arrow[r, "0.64"] & \boxed{7} & \\
\boxed{29} \arrow[dr, "0.44"] & & \boxed{17} \\
& \boxed{26} \arrow[ur, "0.39"] \arrow[dr, "0.46"'] & \\
\boxed{13} \arrow[ur, "0.33"'] & & \boxed{22}
\end{tikzcd}$$

Figure 5: Obtained topics and topic translations by word-level HMM from the English news dataset.

Table 7: Obtained topics by 3-layered MMM from the English news dataset.

	The three most probable clusters		The five most probable words
day cluster z_d	article cluster z_{da}	sentence cluster z_{das}	
3	25	6	prior, record, div, pay, qtly
		19	revs, oper, note, avg, shrs
		29	quarter, earnings, reported, ended, tax
	26	27	management, plant, unit, selling, underwriting
		22	computer, system, products, software, data
		20	owned, subsidiary, unit, plc, agreed
	23	5	common, shareholders, dividend, board, record
		1	offer, proceeds, stake, common, capital
		27	management, plant, unit, selling, underwriting
26	28	16	buffer, strike, minister, party, union
		2	president, chief, executive, chairman, officer
		11	there, think, don't, no, do
	19	26	world, loans, payments, economic, countries
		11	there, think, don't, no, do
		3	tax, growth, rate, budget, deficit
	27	9	industry, companies, financial, markets, business
		28	japan, japanese, minister, president, reagan
		26	world, loans, payments, economic, countries

bines slice sampling and dynamic programming. It would be interesting to extend it for the extended model by introducing our dynamic programming algorithm.

References

[1] Goo blog. `http://blog.goo.ne.jp`.

[2] Reuters-21578 text categorization test collection. `http://www.daviddlewis.com/resources/testcollections/reuters21578/`.

[3] Mark Andrews and Gabriella Vigliocco. The Hidden Markov Topic Model: A Probabilistic Model of Semantic Representation. *Topics in Cognitive Science*, 2(1):101–113, January 2010.

[4] MJ Beal. The infinite hidden Markov model. In *Proc. NIPS*, 2001.

[5] M.J. Beal and Z. Ghahramani. The Variational Bayesian EM Algorithm for Incomplete Data: with Application to Scoring Graphical Model Structures. *Bayesian Statistics*, 7, 2003.

[6] David M Blei, Lawrence Carin, and David Dunson. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, November 2012.

[7] David M Blei, AY Ng, and MI Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[8] DM Chickering. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics*. 1996.

[9] DM Chickering, D Heckerman, and C Meek. Large-sample learning of Bayesian networks is NP-hard. *JMLR*, 5:1287–1330, 2004.

[10] Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, October 1992.

[11] Lan Du, Wray Buntine, and Huidong Jin. A segmented topic model based on the two-parameter Poisson-Dirichlet process. *Machine learning*, 81(1):5–19, July 2010.

[12] Roger Grosse and RR Salakhutdinov. Exploiting compositionality to explore a large space of model structures. In *Proc. UAI*, 2012.

[13] Amit Gruber, Y Weiss, and M Rosen-Zvi. Hidden topic Markov models. In *Proc. AISTATS*, 2007.

[14] Thomas Hofmann. Probabilistic latent semantic analysis. In *Proc. UAI*, 1999.

[15] Tomoharu Iwata, Tsutomu Hirao, and Naonori Ueda. Unsupervised Cluster Matching via Probabilistic Latent Variable Models. In *Proc. AAAI*, 2013.

[16] Tomoharu Iwata, T Yamada, and N Ueda. Probabilistic latent semantic visualization: topic model for visualizing documents. In *Proc. KDD*, 2008.

[17] Changhe Yuan James Cussens, Brandon Malone. Tutorial on Optimal Algorithms for Learning Bayesian Networks. In *IJCAI 2013 Tutorial*, 2013.

[18] Do-kyum Kim, G Voelker, and LK Saul. A Variational Approximation for Topic Modeling of Hierarchical Corpora. In *Proc. ICML*, 2013.

[19] David J C Mackay. Bayesian interpolation. *Neural computation*, 4(3):415–447, May 1992.

[20] Brandon Malone and C Yuan. Evaluating anytime algorithms for learning optimal Bayesian networks. In *Proc. UAI*, 2013.

[21] Thomas P Minka. Estimating a Dirichlet distribution, 2000. `http://research.microsoft.com/minka/papers/dirichlet`.

[22] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, California: Morgan Kaufmann, 1988. 2nd printing edition.

[23] YW Teh and MI Jordan. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[24] Marc Teyssier and D Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proc. UAI*, 2005.

[25] Jurgen Van Gael, Yunus Saatci, Yee Whye Teh, and Zoubin Ghahramani. Beam sampling for the infinite hidden Markov model. In *Proc. ICML*, 2008.

[26] Su-in Lee Varun and Ganapahthi Daphne. Efficient Structure Learning of Markov Networks using L1-Regularization. In *Proc. NIPS*, 2006.

[27] Changhe Yuan, Brandon Malone, and Xiaojian Wu. Learning optimal Bayesian networks using A* search. In *Proc. IJCAI*, 2011.

[28] NL Zhang. Hierarchical latent class models for cluster analysis. *JMLR*, 5:697–723, 2004.