Progressive EM for Latent Tree Models and Hierarchical Topic Detection

Peixian Chen,[†] **Nevin L. Zhang,**[†] **Leonard K. M. Poon,**[‡] **Zhourong Chen**[†]
[†]The Hong Kong University of Science and Technology {pchenac,lzhang,zchenbb@cse.ust.hk}
[‡]The Hong Kong Institute of Education{kmpoon@ied.edu.hk}

Abstract

Hierarchical latent tree analysis (HLTA) is recently proposed as a new method for topic detection. It differs fundamentally from the LDA-based methods in terms of topic definition, topic-document relationship, and learning method. It has been shown to discover significantly more coherent topics and better topic hierarchies. However, HLTA relies on the *Expectation-Maximization* (EM) algorithm for parameter estimation and hence is not efficient enough to deal with large datasets. In this paper, we propose a method to drastically speed up HLTA using a technique inspired by the advances in the method of moments. Empirical experiments show that our method greatly improves the efficiency of HLTA. It is as efficient as the state-of-the-art LDA-based method for hierarchical topic detection and finds substantially better topics and topic hierarchies.

Introduction

Detecting topics and topic hierarchies from document collections, along with its many potential applications, is a major research area in Machine Learning. Currently the predominant approach to topic detection is *latent Dirichlet allocation* (LDA) (Blei, Ng, and Jordan, 2003). LDA has been developed to detect topics and to model relationships among them, including topic correlations (Blei and Lafferty, 2007), topic hierarchies (Blei, Griffiths, and Jordan, 2010; Paisley et al., 2012), and topic evolution (Blei and Lafferty, 2006). We collectively name these methods *LDA-based methods*. In those methods, a topic is a probability distribution over a vocabulary and a document is a mixture of topics. Therefore LDA is a type of *mixed membership model*.

A totally different approach to hierarchical topic detection is recently proposed by Liu, Zhang, and Chen (2014). It is called *hierarchical latent tree analysis* (HLTA), where topics are organized hierarchically as a *latent tree model* (LTM) (Zhang, 2004; Zhang, Wang, and Chen, 2008a) such as the one in Fig 1. Each internal node of an LTM gives several states, with each state being a topic and corresponding to a collection of documents. A document can thus belong to multiple topics from different internal nodes. HLTA is therefore a type of *multiple membership model*.

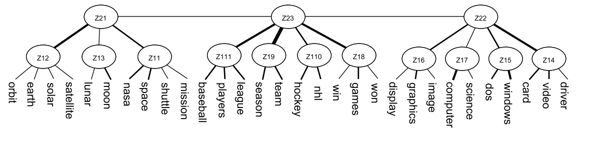


Figure 1: Latent tree model obtained from a toy text dataset.

Empirical results from Liu, Zhang, and Chen (2014) indicate that HLTA finds significantly better topics and topic hierarchies than *hierarchical latent Dirichlet allocation* (hLDA), the first LDA-based method for hierarchical topic detection. However HLTA is not efficient. It took, for instance, 17 hours to process a NIPS dataset that consists of fewer than 2,000 documents over 1,000 distinct words.

The computational bottleneck of HLTA lies in the use of the EM algorithm (Dempster, Laird, and Rubin, 1977) for parameter estimation. In this paper, we propose *progressive EM* (PEM) as a replacement of EM so as to scale up HLTA. PEM is motivated by a spectral technique applied in the method of moments, which relates each equation of population moments to at most 3 observed variables (Chang, 1996; Anandkumar et al., 2012). Similarly, PEM works in steps and, at each step, it focuses on a small part of the model parameters and involves only three or four observed variables.

Our new algorithm is hence named PEM-HLTA. It is drastically faster than HLTA. PEM-HLTA finished processing the aforementioned NIPS dataset within 4 minutes and is capable of analysing much larger corpus. PEM-HLTA is also as efficient as nHDP (Paisley et al., 2012), a state-of-the-art LDA-based method for hierarchical topic detection, and it significantly outperforms nHDP, as well as hLDA, in terms of the quality of topics and topic hierarchies.

Preliminaries

A *latent tree model* (LTM) is a Markov random field over an undirected tree, where the leaf nodes represent observed variables and the internal nodes represent latent variables (Zhang, 2004; Chen et al., 2012). Here we assume all variables are discrete with finite *cardinality*, i.e., finite number of possible states.

Parameters of an LTM consist of potentials associated with edges and nodes such that the product of all potentials is a joint distribution over all variables. We pick the potentials as follows: Root the model at an arbitrary latent node, direct the edges away from the root, and specify a marginal distribution for the root and a conditional distribution for each of the other nodes given its parent. Then in Fig 2(b), if Y is the root, the parameters are the distributions $P(Y)$, $P(A \mid Y), P(Z \mid Y), P(C \mid Z)$ and so forth. Because of the way the potentials are picked, LTMs are technically tree-structured Bayesian networks (Pearl, 1988).

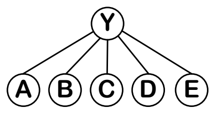
LTMs with a single latent variable are known as *latent class models (LCMs)*(Bartholomew and Knott, 1999). They are a type of finite mixture model for discrete data. For example, the model m_1 in Fig 2(a) defines the following mixture distribution over the observed variables:

$$P(A, \cdots, E) = \sum_{i=1}^{|Y|} P(Y = y_i) P(A, \cdots, E \mid Y = y_i) \quad (1)$$

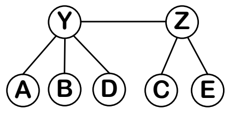
where $|Y|$ is the cardinality of Y and y_i is the value of Y at i^{th} state. If the model is learned from a corpus, then the documents are partitioned into $|Y|$ soft clusters, each represented by a state of Y, and interpreted as a topic. The model m_2 in Fig 2(b) has two latent variables. It gives two different and related mixture distributions:

$$P(A, \cdots, E) = \sum_{i=1}^{|Y|} P(Y = y_i) P(A, \cdots, E \mid Y = y_i),$$
$$P(A, \cdots, E) = \sum_{j=1}^{|Z|} P(Z = z_j) P(A, \cdots, E \mid Z = z_j).$$

The model partitions the corpus in two ways: one into $|Y|$ soft clusters and another into $|Z|$ soft clusters. A document can thus belong to y_i and z_j simultaneously.



(a) m_1



(b) m_2

Figure 2: Leaf nodes are observed while others are latent.

The Algorithm

The input to our PEM-HLTA algorithm is a collection $\mathcal{D}$ of documents, each represented as a binary vector over a vocabulary $\mathcal{V}$. The values in the vector indicate the presence or absence of words in the document. The output is an LTM, where the word variables are at the bottom and the latent variables, all assumed binary, form several levels of hierarchy on top. Topics and topic hierarchy are then extracted from model, as will be explained in empirical results section.

Top Level Control

The top level algorithm for PEM-HLTA is given in Algorithm 1. We briefly illustrate how PEM-HLTA builds a hierarchical LTM from $\mathcal{D}$ using Fig 1, learned from a dataset with 30 word variables. In the first pass through the loop, the subroutine BUILDISLANDS is called (line 3). It partitions all variables into 11 clusters (Fig 3 bottom), which are *uni-dimensional* in the sense that the co-occurrences of words in each cluster can be properly modeled using a single latent variable. A latent variable is introduced for each cluster to form an LCM. We metaphorically refer to the LCMs as islands and the latent variables in them as level-1 latent variables.

Algorithm 1 PEM-HLTA($\mathcal{D}$, τ, δ, κ)

Inputs: $\mathcal{D}$—Collection of documents, τ—Upper bound on the number of top-level topics, δ—Threshold used in UD-test, κ—Number of EM steps on final model.

Outputs: A hierarchical LTM and a topic hierarchy.

```
 1: $\mathcal{D}_1 \leftarrow \mathcal{D}$, $\mathcal{L} \leftarrow \emptyset$, $m \leftarrow null$.
 2: repeat
 3:     $\mathcal{L} \leftarrow$ BUILDISLANDS($\mathcal{D}_1$, $\delta$);
 4:     $m_1 \leftarrow$ BRIDGEISLANDS($\mathcal{L}$, $D_1$);
 5:     if $m = null$ then
 6:         $m \leftarrow m_1$;
 7:     else
 8:         $m \leftarrow$ STACKMODELS($m_1$, $m$);
 9:     end if
10:     $\mathcal{D}_1 \leftarrow$ HARDASSIGNMENT($m$, $\mathcal{D}$);
11: until $|\mathcal{L}| < \tau$.
12: Run EM on $m$ for $\kappa$ steps.
13: return $m$ and topic hierarchy extracted from $m$.
```

The next step is to link up the 11 islands (line 4). This is done by estimating the *mutual information* (MI) (Cover and Thomas, 2012) between every pair of latent variables and building a Chow-Liu tree (Chow and Liu, 1968) over them, so as to form an overall model (Liu et al., 2013). The result is the model in the middle of Fig 3.

In the subroutine HARDASSIGNMENT, inference is carried out to compute the posterior distribution of each latent variable for each document. The document is assigned to the state with the maximum posterior probability, resulting in a dataset over the level-1 latent variables (line 10). In the second pass through the loop, the level-1 latent variables are partitioned into 3 groups. The 3 islands are linked up to form the model shown at the top of Fig 3. At line 8, the model at the top of Fig 3 (m_1) is stacked on the model in the middle (m) to give rise to the final model in Fig 1. While doing so, the subroutine STACKMODELS cuts off the links among the level-1 latent variables. The number of nodes at the top level is below the threshold τ, if we set $\tau = 5$, and hence the loop is exited. EM is run on the final model for κ steps to improve its parameters (line 12). In our experiments, we set $\kappa = 50$. Intuitively, the co-occurrence of words are captured by the level-1 latent variables, whose co-occurring patterns are captured by higher level latent variables. Then a topic hierarchy can be extracted, with topics on top more general and topics at the bottom more specific.

Building Islands

The subroutine BUILDISLANDS(Algorithm 2) starts by calling ONEISLAND to identify a uni-dimensional subset of observed variables and builds an LCM with them, then repeats the process on those observed variables left to create more islands until all variables are included in these islands.

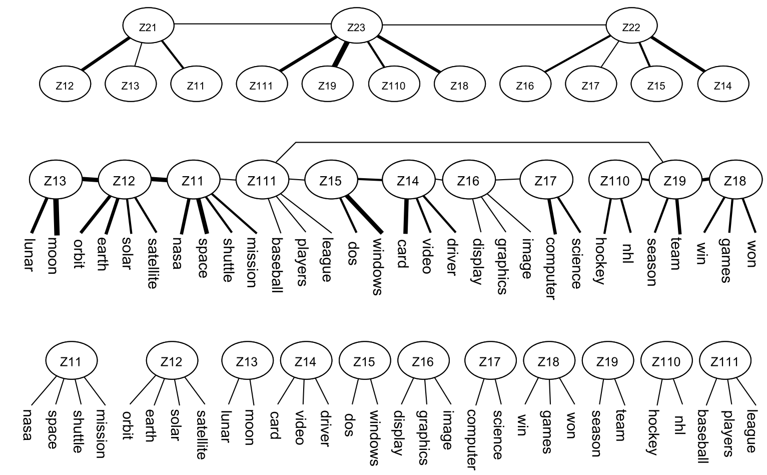


Figure 3: Intermediate models created by PEM-HLTA on a toy dataset. Bottom: Word variables are partitioned into unidimensional clusters (bottom) and a latent variable is introduced for each cluster. Middle: The latent variables are linked up to form a global model and they turned into observed variable using hard-assignment. Top: The process is repeated on the level-1 latent variables. Finally, the model at the top is stacked on the model in the middle to give rise to the model shown in Fig 1.

Algorithm 2 BUILDISLANDS($\mathcal{D}$, δ)

```
1: $\mathcal{V} \leftarrow$ variables in $\mathcal{D}$, $\mathcal{M} \leftarrow \emptyset$.
2: while $|\mathcal{V}| > 0$ do
3:     $m \leftarrow$ ONEISLAND($\mathcal{D}$, $\mathcal{V}$, $\delta$);
4:     $\mathcal{M} \leftarrow \mathcal{M} \cup \{m\}$;
5:     $\mathcal{V} \leftarrow$ variables in $\mathcal{D}$ but not in any $m \in \mathcal{M}$;
6: end while
7: return $\mathcal{M}$.
```

Algorithm 3 ONEISLAND($\mathcal{D}$, $\mathcal{V}$, δ)

```
1: if $|\mathcal{V}| \leq 3$, $m \leftarrow$ LEARNLCM($\mathcal{D}$, $\mathcal{V}$), return $m$.
2: $\mathcal{S} \leftarrow$ three variables in $\mathcal{V}$ with highest MI,
3: $\mathcal{V}_1 \leftarrow \mathcal{V} \setminus \mathcal{S}$;
4: $\mathcal{D}_1 \leftarrow$ PROJECTDATA($\mathcal{D}$, $\mathcal{S}$),
5: $m \leftarrow$ LEARNLCM($\mathcal{D}_1$, $\mathcal{S}$).
6: loop
7:     $X \leftarrow \arg\max_{A \in \mathcal{V}_1} MI(A, \mathcal{S})$,
8:     $W \leftarrow \arg\max_{A \in S} MI(A, X)$,
9:     $\mathcal{D}_1 \leftarrow$ PROJECTDATA$(\mathcal{D}, \mathcal{S} \cup \{X\})$, $\mathcal{V}_1 \leftarrow \mathcal{V}_1 \setminus \{X\}$.
10:    $m_1 \leftarrow$ PEM-LCM$(m, \mathcal{S}, X, \mathcal{D}_1)$.
11:    if $|\mathcal{V}_1| = 0$, return $m_1$.
12:    $m_2 \leftarrow$ PEM-LTM-2L($m$, $\mathcal{S} \setminus \{W\}$, $\{W, X\}$, $\mathcal{D}_1$)
13:    if $BIC(m_2|\mathcal{D}_1) - BIC(m_1|\mathcal{D}_1) > \delta$ then.
14:        return $m_2$ with $W$, $X$ and their parent removed.
15:    end if
16:    $m \leftarrow m_1, \mathcal{S} \leftarrow \mathcal{S} \cup \{X\}$.
17: end loop
```

Uni-Dimensionality Test We rely on the *uni-dimensionality test (UD-test)* (Liu et al., 2013) to determine whether a set $\mathcal{S}$ of variables is uni-dimensional. The idea is to compare two LTMs m_1 and m_2, where m_1 is the best model among all LCMs for $\mathcal{S}$ while m_2 is the best one among all LTMs with two latent variables. The model selection criterion used is the BIC score (Schwarz, 1978). The set $\mathcal{S}$ is uni-dimensional if the following inequality holds:

$$\mathcal{BIC}(m_2 \mid \mathcal{D}) - \mathcal{BIC}(m_1 \mid \mathcal{D}) < \delta, \qquad (2)$$

where δ is a threshold. It means $\mathcal{S}$ is considered uni-dimensional if the best two-latent variable model is not significantly better than the best one-latent variable model. The quantity on the left of Equation (2) is a large sample approximation of the natural logarithm of Bayes factor (Raftery, 1995) for comparing m_1 and m_2. Based on the cut-off values for the Bayes factor, we set $\delta = 3$ in our experiments.

Building An Island ONEISLAND identifies a uni-dimensional subset from given variables $\mathcal{V}$ and builds an LCM. We illustrate this process using Fig 2. Define the mutual information between a variable Z and a set $\mathcal{S}$ as $\text{MI}(Z, \mathcal{S}) = \max_{A \in \mathcal{S}} \text{MI}(Z, A)$. ONEISLAND maintains a working set $\mathcal{S}$ of observed variables. Initially, $\mathcal{S}$ contains the pair of variables with the highest MI and a third variable that has the highest MI with the pair (line 2), say A, B and C. Then an LCM is learned for those three variables using the subroutine LEARNLCM. Then other variables are added to $\mathcal{S}$ one by one until the UD-test fails. Let D be the variable that has the maximum MI with $\mathcal{S}$ among all other variables. Suppose the UD-test passes on $\mathcal{S} \cup \{D\}$, then D is added to $\mathcal{S}$. Next let E be the variable with the maximum MI with S (line 7) and the UD-test is performed on $S \cup E = \{A, B, C, D, E\}$ (lines 8-14). The two models m_1 and m_2 used in the test is shown in Fig 2. For computational efficiency, we do not search for the best structure for m_2. Instead, the structure is determined as follows: Pick the variable in $\mathcal{S}$ that has the maximum MI with E (line 8) (let it be C), and group it with E in the model (line 12). The model parameters are estimated using the subroutines PEM-LCM and PEM-LTM-2L, which will be explained in the next section. If the test fails, then C, E and Z are removed from m_2, and what remains in the model, an LCM, is returned. If the test passes, E is added to $\mathcal{S}$ (line 16) and the process continues.

PEM for Model Construction

PEM-HLTA conceptually consists of a model construction phase (lines 2-11) and a parameter estimation phase (line 12). During the first phase, many intermediate models are constructed. In this paper, we propose a fast method for estimating the parameters of those intermediate models.

A Spectral Technique We begin by presenting a property of LTMs that motivates our new method. A similar property on evolutionary trees was first discovered by Chang (1996), and Zhang, Wang, and Chen (2014) derived it in the context of LTMs. We introduce some notation using m_1 of Fig 2. Since all variables have the same cardinality, the conditional distribution $P(A|Y)$ can be regarded as a square matrix denoted as $P_{A|Y}$. Similarly, P_{AC} is the matrix representation of the joint distribution $P(A,C)$. For a value b of B, $P_{b|Y}$ is the vector presentation of $P(B{=}b|Y)$ and P_{AbC} the matrix representation of $P(A,B{=}b,C)$.

Theorem 1 ***[Zhang, Wang, and Chen (2014)]*** *Let Y be the latent variable in an LCM and A,B,C be three of the observed variables. Assume all variables have the same cardinality and the matrices $P_{A|Y}$ and P_{AC} are invertible. Then we have*

$$P_{A|Y}\,\mathrm{diag}(P_{b|Y})P_{A|Y}^{-1} = P_{AbC}P_{AC}^{-1}, \tag{3}$$

where $\mathrm{diag}(P_{b|Y})$ is a diagonal matrix with components of $P_{b|Y}$ as the diagonal elements.

The equation implies that the model parameters $P(B{=}b|Y{=}0),\cdots,P(B{=}b|Y{=}|Y|)$ are the eigenvalues of the matrix on the right, and hence can be obtained from the marginal distributions P_{AbC} and P_{AC}.

In the method of moments, Theorem 1 can be used to estimate $P(B|Y)$ under two conditions: (1) There is a good fit between the data and model, and (2) the sample size is sufficiently large. In this case, the empirical marginal distributions $\hat{P}(A,B,C)$ and $\hat{P}(A,C)$ computed from data are accurate estimates of the distributions $P(A,B,C)$ and $P(A,C)$ of the model. They can be used to form the matrix $P_{AbC}P_{AC}^{-1}$, and to determine $P_{B|Y}$ as the eigenvalues of the matrix. Theorem 1 still applies when replacing edges like (Y,A) with paths. For example in Fig 2(b), if $P(C|Z)$ and $P(E|Z)$ are to be estimated, a third observed variable can be chosen from (A,B,D) as long as there is path from Z to this observed variable.

Theorem 1 can be also used to estimate all the parameters of the model in Fig 2(a). First, we estimate $P(B|Y)$ using Equation 3 in the sub-model Y-$\{A,B,C\}$. By swapping the roles of variables, we obtain $P(A|Y)$ and $P(C|Y)$. Next we can consider the sub-model Y-$\{B,C,D\}$ and estimate $P(D|Y)$ with $P(B|Y)$ and $P(C|Y)$ fixed. And we do the same with the variables left. The parameters are then estimated in steps instead of all at once. Hence we call this scheme *progressive parameter estimation*.

Progressive EM By solving equations of lower-order moments, the method of moments can be drastically faster than EM. Unfortunately, it does not produce high quality estimates when the model does not fit data well and/or the sample size is not sufficiently large, as the empirical marginal distributions $\hat{P}(A,B,C)$ and $\hat{P}(A,C)$ are poor estimates of true distributions. In our experiences, the method frequently gives negative estimates for probability values in the context of latent tree models.

In this paper, we do not estimate parameters by solving the equation in Theorem 1. However, we adopt a progressive scheme combined with EM. This gives rise to *progressive EM (PEM)*. To estimate the parameters of m_1, PEM first estimates $P(Y)$, $P(A|Y)$, $P(B|Y)$, and $P(C|Y)$ by running EM on the sub-model Y-$\{A,B,C\}$; then it estimates $P(D|Y)$ by running EM on the sub-model Y-$\{B,C,D\}$ with $P(B|Y)$, $P(C|Y)$ and $P(Y)$ fixed and so forth. All the sub-models involve 3 observed variables. This is exactly the idea behind subroutine LEARNLCM.

For m_2, PEM first runs EM on sub-model Y-$\{A,B,D\}$; then it estimates $P(C|Z)$, $P(E|Z)$ and $P(Z|Y)$ by running EM on the two latent variable sub-model $\{B,D\}$-Y-Z-$\{C,E\}$, with $P(B|Y)$, $P(D|Y)$ and $P(Y)$ fixed. Note that only two of the children of Y are used here, and the model involves only 4 observed variables.

Intuitively, the method of moments tries to fit data in a rigid way, while PEM tries to fit data in an elastic manner. It never gives negative probability values. Moreover, it is still efficient because EM is run only on sub-models with three or four observed binary variables, and multiple starting points can be tried to alleviate the issues of local maxima with much lower computational cost.

PEM for Island Building PEM can be aligned with the subroutine ONEISLAND nicely because the subroutine adds variables to the working set $\mathcal{S}$ one at a time. Consider a pass through the loop. At the beginning, we have an LCM m for the variables in $\mathcal{S}$, whose parameters have been estimated earlier. Then ONEISLAND finds the variable X outside $\mathcal{S}$ that has the maximum MI with $\mathcal{S}$, and the variable W inside $\mathcal{S}$ that has the maximum MI with X (line 7, 8).

At line 11, ONEISLAND adds X to m to create a new LCM m_1, and estimates the parameters for the new variable using the subroutine PEM-LCM. We illustrate how this is done using Fig 2. Suppose the LCM m is the model Y-$\{A,B,C,D\}$ and the variable X is E. PEM-LCM adds the variable E to m and thereby creates a new LCM m_1, which is Y-$\{A,B,C,D,E\}$ (Fig 2(a)). To estimate $P(E|Y)$, PEM-LCM creates a temporary model m' from m_1 by only keeping three observed variables: E and two other variables with maximum MI with E. Suppose m' is Y-$\{C,D,E\}$. PEM-LCM estimates the distribution $P(E|Y)$ by running EM on m' with all other parameters fixed. Finally, it copies $P(E|Y)$ from m' to m_1, and returns m_1.

At line 12, ONEISLAND adds X to m and learns a two-latent variable model m_2 using the subroutine PEM-LTM-2L. We illustrate PEM-LTM-2L using the foregoing example. Let X be E and W be C. PEM-LTM-2L cre-

Table 1: Parts of the topic hierarchies obtained by nHDP (left) and PEM-HLTA (right) on Nips-10k.

1. gaussian likelihood mixture density Bayesian	**1. [0.22] mixture gaussian mixtures em covariance**
1.1. gaussian density likelihood Bayesian	1.1. [0.23] em maximization ghahramani expectation
1.2. frey hidden posterior chaining log	1.2. [0.23] mixture gaussian mixtures covariance
1.3. classifier classifiers confidence	1.3. [0.23] generative dis generafive generarive
1.4. smola adaboost onoda mika svms	1.4. [0.27] variance noise exp variances deviation
1.5. speech context hme hmm experts	1.4.1. [0.28] variance exp variances deviation cr
	1.4.2. [0.44] noise noisy robust robustness mean
2. image recognition images feature features	**2. [0.26] images image pixel pixels object**
2.1. image recognition feature images object	2.1. [0.25] images image features detection face
2.2. smola adaboost onoda utterance	2.2. [0.24] camera video imaging false tracked
2.3. object matching shape image features	2.3. [0.24] pixel pixels intensity intensities
2.4. nearest basis examples rbf classifier	2.4. [0.17] object objects shape views plane
2.5. tangent distance simard distances	2.5. [0.20] rotation invariant translation
	2.6. [0.26] nearest neighbor kohonen neighbors
3. rules language rule sequence context	**3. [0.15] speech word speaker language phoneme**
3.1. recognition speech mlp word trained	3.1. [0.16] word language vocabulary words sequence
3.2. rules rule stack machine examples	3.2. [0.11] spoken acoustics utterances speakers
3.3. voicing syllable fault faults units	3.3. [0.10] string strings grammar symbol symbols
3.4. rules hint table hidden structure	3.4. [0.06] retrieval search semantic searching
3.5. syllable stress nucleus heavy bit	3.5. [0.14] phoneme phonetic phonemes waibel lang
	3.6. [0.15] speech speaker acoustic hmm hmms

ates the new model m_2, which is $\{A, B, D\}$-Y-Z-$\{C, E\}$ (Fig 2(b)). To estimate the parameters $P(C|Z)$, $P(E|Z)$ and $P(Z|Y)$, PEM-LTM-2L creates a temporary model m' which is $\{A, D\}$-Y-Z-$\{C, E\}$. Only the two of the children of Y that have maximum MI with E remain(A and D). PEM-LTM-2L estimates the three distributions by running EM on m' with all other parameters fixed. Finally, it copies the distributions from m' to m_2 and returns m_2. Similarly in the subroutine BRIDGEDISLANDS we use this method to estimate parameters between latent variables, but only estimating $P(Z|Y)$ and keeping all other parameters fixed.

Empirical results

We aim at scaling up HLTA, hence we need to empirically determine how efficient PEM-HLTA is compared with HLTA. We also compare PEM-HLTA with nHDP, the state-of-the-art LDA-based method for hierarchical topic detection, in terms of computational efficiency and quality of results. Also included in the comparisons are hLDA and a method named CorEx (Ver Steeg and Galstyan, 2014) that builds hierarchical latent trees by optimizing an information-theoretic objective function.

Two of the datasets used are NIPS[1] and Newsgroup[2]. Three versions of the NIPS data with vocabulary sizes 1,000, 5,000 and 10,000 were created by choosing words with highest average TF-IDF values, referred to as Nips-1k, Nips-5k and Nips-10k. Similarly, two versions (News-1k and News-5k) of the Newsgroup data were created. Note that News-10k is not included because it is beyond the capabilities of three of the methods. Comparisons of PEM-HLTA and nHDP on large data will be given later. After preprocessing, NIPS and Newsgroup consist of 1,955 and 19,940 documents respectively. For PEM-HLTA, HLTA and CorEx, the data are represented as binary vectors, whereas for nHDP and hLDA, they are represented as bags-of-words.

PEM-HLTA determines the height of hierarchy and the number of nodes at each level automatically. On the NIPS and Newsgroup datasets, it produces hierarchies with between 4 to 6 levels. For nHDP and hLDA, the height of hierarchy needs to be manually set and is usually set at 3. We set the number of nodes at each level in such way that nHDP and hLDA would yield roughly the same total number of topics as PEM-HLTA. CorEx is configured similarly. PEM-HLTA is implemented in Java. The parameter settings are described along algorithm description. Implementations of other algorithms are provided by their authors and ran at their default parameter settings. All experiments are conducted on the same desktop computer.[3]

Topic Hierarchies for Nips-10k

Table 1 shows parts of the topic hierarchies obtained by nHDP and PEM-HLTA. The left half displays 3 top-level topics by nHDP and their children. Each nHDP topic is represented using the top 5 words occurring with highest probabilities in the topic. The right half show 3 top-level topics yielded by PEM-HLTA and their children. The topics are extracted from the model learned by PEM-HLTA as follows: For a latent binary variable Z in the model, we enumerate the word variables in the sub-tree rooted at Z in descending order of their MI values with Z. The leading words are those whose probabilities differ the most between the two states of Z and are hence used to characterize the states. The state of Z under which the words occur less often overall is regarded as the *background topic* and is not reported, while the other state is reported as a genuine topic. Values in [] show the percentage of the documents belonging to the genuine topic.

[1]http://www.cs.nyu.edu/ roweis/data.html

[2]http://qwone.com/jason/20Newsgroups/

[3]The code for HLTA algorithm can be found at http://www.cse.ust.hk/∼lzhang/topic/index.htm, along with the datasets used in the papers and the results.

Let us examine some of the topics. We refer to topics on the left using the letter L followed by topic numbers and those on the right using R. For PEM-HLTA, R1 consists of probability terms: R1.1 is about EM algorithm; R1.2 about Gaussian mixtures and R1.3 about generative distributions. R1.4 is a combination of variance and noise, which are separated at the next lower level. For nHDP, the topic L1 and its children L1.1, L1.2 and L1.5 are also about probability. However, L1.3 and L1.4 do not fit in the group well.

The topic R2 is about image analysis, while its first four subtopics are about different aspects of image analysis: *sources of images, pixels, objects*. R2.5 and R2.6 are also meaningful and related, but do not fit in well. They are placed in another subgroup by PEM-HLTA. In nHDP, the subtopics of L2 do not give a clear spectrum of aspects of image analysis. The topic R3 is about speech recognition. Its subtopics are about different aspects of speech recognition. Only R3.4 does not fit in the group well. In contrast, L3 and its subtopics do not present a clear semantic hierarchy. Some of them are not meaningful. Another topic related to speech recognition L1.5 is placed elsewhere. Overall, the topics and topic hierarchy obtained by PEM-HLTA are more meaningful than those by nHDP.

Topic Coherence and Model Quality

To quantitatively measure the quality of topics, we use the *topic coherence score* proposed by Mimno et al. (2011). The metric depends on the number M of words used to characterize a topic. We set $M = 4$. Then held-out likelihood is used to assess the quality of the models. Each dataset was randomly partitioned into a training set with 80% of the data, and a test set with the 20% left.

Table 2 shows the average topic coherence scores of the topics produced by the five algorithms. The sign "-" indicates running time exceeded 72 hours. The quality of topics produced by PEM-HLTA is similar to those by HLTA on Nips-1k and News-1k, and better on Nips-5k. In all cases, PEM-HLTA produced significantly better topics than nHDP and the other two algorithms. The held-out per-document loglikelihood statistics are shown in Table 3. The likelihood values of PEM-HLTA are similar to those of HLTA, showing that the use of PEM to replace EM does not influence model quality much. They are significantly higher than those of CorEx. Note that the likelihood values in Table 3 for the LDA-based methods are calculated from bag-of-words data. They are still lower than the other methods even calculated from the same binary data as for the other three methods. It should be noted that, in general, better model fit does not necessarily imply better topic quality (Chang et al., 2009). In context of hierarchical topic detection, however, PEM-HLTA not only leads to better model fit, but also gives better topics and better topic hierarchies.

Running times

Table 4 shows the running time statistics. PEM-HLTA drastically outperforms HLTA, and the difference increases with vocabulary size. On Nips-10k and News-5k, HLTA did not terminate in 3 days, while PEM-HLTA finished the computation in about 6 hours. PEM-HLTA is also faster than nHDP,

Table 2: Average topic coherence scores.

	Nips-1k	Nips-5k	Nips-10k	News-1k	News-5k
PEM-HLTA	-6.25	**-8.04**	**-8.87**	-12.30	**-13.07**
HLTA	**-6.23**	-9.23	—	**-12.08**	—
hLDA	-6.99	-8.94	—	—	—
nHDP	-8.08	-9.55	-9.86	-14.26	-14.51
CorEx	-7.23	-9.85	-10.64	-13.47	-14.51

Table 3: Per-document loglikelihood

	Nips-1k	Nips-5k	Nips-10k	News-1k	News-5k
PEM-HLTA	**-390**	**-1,117**	**-1,424**	**-116**	**-262**
HLTA	-391	-1,161	—	-120	—
hLDA	-1,520	-2,854	—	—	—
nHDP	-3,196	-6,993	-8,262	-265	-599
CorEx	-442	-1,226	-1,549	-140	-322

although the difference decreases with vocabulary size as nHDP works in a stochastic way (Paisley et al., 2012). Moreover, PEM-HLTA is more efficient than hLDA and CorEx.

Stochastic EM

Conceptually PEM-HLTA has two phases: hierarchical model construction and parameter estimation. In the second phase, EM is run a predefined number of steps from the initial parameter values from the first phase. It is time-consuming if the sample size is large. Paisley et al. (2012) faced a similar problem with nHDP. They solve the problem using stochastic inference. We adopt the same idea for the second phase of PEM-HLTA and call it *stochastic EM*. We sample the whole dataset into subsets and process the subsets one by one. Model parameters are updated after processing each data subset and overall one goes through the entire dataset. We tested the idea on the New York Times dataset[4], which consists of 300,000 articles. To analyze the data, we picked 10,000 words using TF-IDF and then randomly sampled the dataset into 50 smaller but equal-sized subsets with each containing 6,000 articles. We used only one subset for the first phase of PEM-HLTA. For the second phase, we ran EM on current model once using each subset in turn until all the subsets are utilized.

On New York Times data, we only compare PEM-HLTA with nHDP since other methods are not amenable to processing large datasets as we can observe from Table 4. We still trained nHDP model using documents in bag-of-words form and PEM-HLTA using documents as binary vectors of words. Table 5 reports the running times and topic coherence. PEM-HLTA took around 11 hours which is a little bit slower than nHDP (10.5 hours). However, PEM-HLTA produced more coherent topics, which is not only testified by the coherence score, but also the resulting topic hierarchies.

Conclusions

We have proposed and investigated a method to scale up HLTA — a newly emerged method for hierarchical topic detection. The key idea is to replace EM using progressive EM.

[4]http://archive.ics.uci.edu/ml/datasets/Bag+of+Words

Table 4: Running times.

Time(min)	Nips-1k	Nips-5k	Nips-10k	News-1k	News-5k
PEM-HLTA	**4**	**140**	**340**	**47**	**365**
HLTA	42	2,020	—	279	—
hLDA	2,454	4,039	—	—	—
nHDP	359	382	435	403	477
CorEx	43	366	704	722	4,025

Table 5: Performances on the New York Times data.

	Time (min)	Average topic coherence
PEM-HLTA	670	-12.86
hHDP	637	-13.35

The resulting algorithm PEM-HLTA reduces the computation time of HLTA drastically and can handle much larger datasets. More importantly, it outperforms nHDP, the state-of-the-art LDA-based method for hierarchical topic detection, in terms of both quality of topics and topic hierarchy, with comparable speed on large-scale data. Though we only showed how PEM works in HLTA, it can possibly be used in other more general models. PEM-HLTA can also be further scaled up through parallelization and used for text classification. We plan to investigate these directions in the future.

Acknowledgment

We thank John Paisley for sharing the nHDP implements with us, and we thank Tengfei Liu, Farhan Khawar and Jun Zhu for valuable discussions. Research on this article was supported by Hong Kong Research Grants Council under grant 16202515 and the Hong Kong Institute of Education under project RG90/2014-2015R.

References

Anandkumar, A., Chaudhuri, K., Hsu, D., Kakade, S. M., Song, L., and Zhang, T. 2012. Spectral methods for learning multivariate latent tree structure. In *Advances in Neural Information Processing Systems*, 2025–2033.

Bartholomew, D. J., and Knott, M. 1999. *Latent Variable Models and Factor Analysis*. Arnold, 2nd edition.

Blei, D. M., and Lafferty, J. D. 2006. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, 113–120. ACM.

Blei, D. M., and Lafferty, J. D. 2007. A correlated topic model of science. *The Annals of Applied Statistics* 17–35.

Blei, D. M., Griffiths, T. L., and Jordan, M. I. 2010. The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. *Journal of the ACM* 57(2):7:1–7:30.

Blei, D. M., Ng, A. Y., and Jordan, M. I. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research* 3:993–1022.

Chang, J., Boyd-Graber, J. L., Gerrish, S., Wang, C., and Blei, D. M. 2009. Reading tea leaves: How humans interpret topic models. In *Advances in Neural Information Processing Systems*, volume 22, 288–296.

Chang, J. T. 1996. Full reconstruction of Markov models on evolutionary trees: Identifiability and consistency. *Mathematical Biosciences* 137(1):51–73.

Chen, T., Zhang, N. L., Liu, T., Poon, K. M., and Wang, Y. 2012. Model-based multidimensional clustering of categorical data. *Artificial Intelligence* 176:2246–2269.

Chow, C. K., and Liu, C. N. 1968. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory* 14(3):462–467.

Cover, T. M., and Thomas, J. A. 2012. *Elements of information theory*. John Wiley & Sons.

Dempster, A. P., Laird, N. M., and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* 39(1):1–38.

Liu, T., Zhang, N. L., Chen, P., Liu, A. H., Poon, L. K., and Wang, Y. 2013. Greedy learning of latent tree models for multidimensional clustering. *Machine Learning* 98(1–2):301–330.

Liu, T., Zhang, N. L., and Chen, P. 2014. Hierarchical latent tree analysis for topic detection. In *Machine Learning and Knowledge Discovery in Databases*, 256–272.

Mimno, D., Wallach, H. M., Talley, E., Leenders, M., and McCallum, A. 2011. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 262–272. Association for Computational Linguistics.

Paisley, J., Wang, C., Blei, D. M., and Jordan, M. I. 2012. Nested hierarchical dirichlet processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, California: Morgan Kaufmann Publishers.

Raftery, A. E. 1995. Bayesian model selection in social research. *Sociological Methodology* 25:111–163.

Schwarz, G. 1978. Estimating the dimension of a model. *The Annals of Statistics* 6(2):461–464.

Ver Steeg, G., and Galstyan, A. 2014. Discovering structure in high-dimensional data through correlation explanation. In *Advances in Neural Information Processing Systems 27*, 577–585.

Zhang, N. L., Wang, Y., and Chen, T. 2008a. Discovery of latent structures: Experience with the CoIL challenge 2000 data set. *Journal of Systems Science and Complexity* 21:172–183.

Zhang, N. L., Wang, Y., and Chen, T. 2008b. Latent tree models and multidimensional clustering of categorical data. Technical Report HKUST-CS08-02, The Hong Kong Univeristy of Science and Technology.

Zhang, N. L., Wang, X., and Chen, P. 2014. A study of recently discovered equalities about latent tree models using inverse edges. In *Probabilistic Graphical Models*, 567–580.

Zhang, N. L. 2004. Hierarchical latent class models for cluster analysis. *Journal of Machine Learning Research* 5:697–723.