Efficient Model Evaluation in the Search-Based Approach to Latent Structure Discovery

Tao Chen, Nevin L. Zhang and Yi Wang
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
Kowloon, Hong Kong, China

Abstract

Latent tree (LT) models are tree-structured Bayesian networks where leaf nodes are observed while internal nodes are hidden. We are interested in learning LT models through systematic search. A key problem here is how to efficiently evaluate candidate models during search. The problem is difficult because there is a large number of candidate models, the candidate models contain latent variables, and some of those latent variables are foreign to the current model. A variety of ideas for attacking the problem have emerged from the literature. In this paper we observe that the ideas can be grouped into two distinct approaches. The first is based on data completion, while the second is based on what we call maximum restricted likelihood. We investigate and compare the two approaches in the framework of EAST, a newly developed search procedure for learning LT models.

1 Introduction

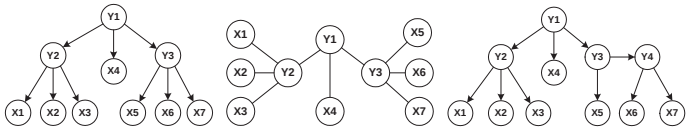
Learning Bayesian networks (BNs) from data has been the focus of much research over the past two decades. Deep insights have been gained for the case where all variables are observed (e.g. Chickering 2002). However, relatively little progress has been made for the case with latent variables. This is not due to the lack of interest on the problem, but its difficulty.

Imagine a search-based algorithm for learning BNs with latent variables. At each step, the algorithm would evaluate a potentially large number of candidate models. Assume the BIC score is used for model selection.[1] To calculate the BIC score of a candidate model m', one needs to compute its maximum loglikelihood $\max_{\theta'} \log P(\mathcal{D}|m', \theta')$, where $\mathcal{D}$ is the data set and θ' is the parameter vector for m'. This requires the expectation-maximization (EM) algorithm. The difficulty is that running EM on a large number of candidate models is computationally prohibitive.

[1]The BICe score suggested by Geiger *et al.* (1996) is currently impractical due to the lack of efficient methods for computing effective model dimension.

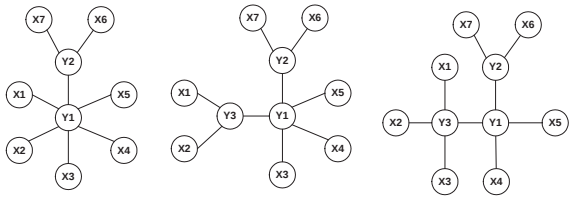
Two methods for overcoming the difficulty have emerged from the literature. The first method is based on the idea of data completion. It first completes the data set $\mathcal{D}$ based on the current model m and uses the completed data set $\bar{\mathcal{D}}$ to evaluate the candidate models. When all the variables in a candidate model m' are observed with respect to $\bar{\mathcal{D}}$, one can evaluate the model using the maximum expected loglikelihood $\max_{\theta'} \log P(\bar{\mathcal{D}}|m', \theta')$ (Friedman 1997). When a candidate model contains variables that are not observed with respect to $\bar{\mathcal{D}}$, one can evaluate the model using heuristics that are computed from $\bar{\mathcal{D}}$ (Zhang and Kočka 2004, Elidan and Friedman 2005).

The second method is based on what we call maximum restricted likelihood. A candidate model m' typically shares many parameters with the current model m. Suppose we have computed the maximum likelihood estimation (MLE) of the parameters of m. Let θ_1^* be the MLE for the parameters that m has in common with m'. Let δ_2 be the parameters of m' that are not shared with m. The method approximates $\max_{\theta'} \log P(\mathcal{D}|m', \theta')$ us-



(a) m (rooted) (b) m (unrooted) (c) m'

Figure 1: Rooted and unrooted latent tree models.



m_1 m_2 m_3

Figure 2: The NI and NR operators.

ing the *maximum restricted loglikelihood*, i.e. $\max_{\delta_2} \log P(\mathcal{D}|m', \theta_1^*, \delta_2)$, and uses the latter to evaluate m'. This method is implicitly used in (Zhang and Kočka 2004). Similar ideas are used in, among others, phylogenetic tree reconstruction (Guindon and Gascuel 2003) and learning of continuous Bayesian networks (Nachman *et al.* 2004).

In this paper we investigate and compare the two methods in the context of latent tree models. Called *hierarchical latent class models* previously (Zhang 2004), *latent tree (LT) models* are tree-structured Bayesian networks where variables at leaf nodes are observed and are hence called *manifest variables*, while variables at internal nodes are hidden and hence are called *latent variables*. All variables are assumed discrete. Figure 1 (a) shows the structure of an LT model.

LT models are interesting for three reasons. First, they relax the local independence assumption of latent class (LC) models (Lazarsfeld and Henry 1968) and hence offer a more general framework for cluster analysis of categorical data (Zhang 2004). Second, LT analysis can reveal latent structures behind data. In this sense LT analysis is a generalization of phylogenetic tree reconstruction (Durbin *et al.* 1998). Using LT models, researchers have found interesting latent structures from stocks data (Elidan and Friedman 2005), marketing data (Zhang 2007) and medical data (Zhang *et al.* 2008). Third, LT models are computationally simple to handle and at the same time can model complex interactions among manifest variables (Pearl 1988). Those two properties make LT models a good tool for density estimation for discrete variables.

2 Search for Optimal Model

Suppose there is a data set $\mathcal{D}$ on a set of manifest variables. To *learn an LT model* from $\mathcal{D}$ means to find a model m that maximizes the BIC score:

$$BIC(m|\mathcal{D}) = \max_{\theta} \log P(\mathcal{D}|m, \theta) - \frac{d(m)}{2} \log N,$$

where θ denotes the set of model parameters, $d(m)$ the number of independent parameters, and N the sample size. It has been shown that edge orientations cannot be determined from data (Zhang 2004). So we can learn only *unrooted latent tree models*, which are latent tree models with all directions on the edges dropped. An example is given in Figure 1 (b). From now on when we speak of latent tree models we always mean unrooted latent tree models unless it is explicitly stated otherwise.

In this section we present a search procedure for learning LT models. Called EAST, the procedure uses five search operators, namely *state introduction (SI)*, *node introduction (NI)*, *node relocation (NR)*, *state deletion (SD)*, and *node deletion (ND)*. They are borrowed from (Zhang and Kočka 2004) with minor modifications. Given an LT model and a latent variable in the model, the *SI* operator creates a new model by adding a new state to the domain of the variable. The *SD* operator does the opposite. The *NI* operator involves one latent node Y and two of its neighbors. It creates a new model by introducing a new latent node Y' to mediate the latent variable and the two neighbors. The cardinality of Y' is set to be that of Y. In m_1 of Figure 2, introducing a new node Y_3 to mediate Y_1 and its neighbors X_1 and X_2 results in m_2. For the sake of computational efficiency, we do not consider introducing a new node to mediate Y and more than two of its

neighbors. The *ND* operator is the opposite of NI. The *NR* operator involves two latent nodes Y_1 and Y_2 and a neighbor Z of Y_1. It creates a new model by relocating Z to Y_2, i.e. removing the link between Z and Y_1 and adding a link between Z and Y_2. In m_2 of Figure 2, relocating X_3 from Y_1 to Y_3 results in m_3.

The search operators can be divide into three groups. NI and SI make the current model more complex and hence are *expansion operators*. ND and SD make the current model simpler and hence are *simplification operators*. NR rearranges connections between the variables and hence is an *adjustment operator*.

The BIC score consists of two terms. The first term measures model fit while the second term penalizes model complexity. Our objective is to optimize the BIC score. Suppose we start with a model that does not fit the data at all. Then improving model fit is the first priority at the initial stage of search. So we first improve model fit by searching with the expansion operators. When the BIC ceases to increase, we switch to the simplification operators in order to minimize the penalty term. The process repeats itself until model score ceases to increase. This is similar to the idea behind the greedy equivalence search (GES) algorithm (Chickering 2002) for learning Bayesian networks.

In the following, we introduce several modifications to the above simple scheme and eventually obtain the EAST procedure given in Figure 3. To understand the first modification, consider the three models in Figure 2. Due to the constraint imposed on NI, it is impossible to reach m_3 directly from m_1. A natural remedy is to consider node relocations after each application of NI. Suppose we have just applied NI to m_1 and have obtained m_2. What we do next is to consider repeatedly relocating the other neighbors of Y_1 in m_1, i.e. X_3, X_4, X_5 and Y_2, to the new latent variable Y_3. This is the `localAdjust`$(m_2, m_1, \mathcal{D})$ subroutine.

The second modification is about choosing between candidate models generated by NI and SI. Let m be the current model and m' be a candidate model. Define the *improvement ratio of* m' *over* m to be

$$IR(m', m|\mathcal{D}) = \frac{BIC(m'|\mathcal{D}) - BIC(m|\mathcal{D})}{d(m') - d(m)}.$$

It is the increase in model score per unit increase in model complexity. The *cost-effectiveness* principle (Zhang and Kočka 2004) states that, among all candidate models generated by SI and NI, choose the one that has the highest improvement ratio.

We have now completed explaining the `expand` subroutine in Figure 3. The other subroutines are more or less straightforward. The `simplify` subroutine first repeatedly applies SD to the current model until the BIC score ceases to increase and then it does the same with ND. The `adjust` subroutine repeatedly applies the NR operator to the current model until it is no longer beneficial to do so. Unlike in `localAdjust`, there is no restriction on the NR operator here. One can relocate a node to anywhere in the current model. This is an effective mechanism to avoid local maxima.

Located at the top of Figure 3 is the EAST procedure itself. The name EAST is a shorthand for *Expansion, Adjustment, Simplification until Termination*. The procedure takes a data set and an initial model as inputs. It first runs the `expand` subroutine on the initial model. Then it `adjust`s connections between nodes. Thereafter, it passes the resultant model to the `simplify` subroutine. If model score is improved in any of the three steps, the entire process repeats itself with the best model obtained as the initial model.

EAST is similar to the search procedure described in (Zhang and Kočka 2004). There are two main differences. The latter groups NR with NI and SI and hence does not have a separate model adjustment phase. It also restricts how far one can relocate a node.

3 Model Evaluation based on Restricted Likelihood

Each of the arg max operators in EAST evaluates a potentially large number of candidate models. In this section we describe the restricted likelihood method for doing this efficiently.

`EAST`$(m, \mathcal{D})$
Repeat until termination:
  $m_1 \leftarrow$ `expand`$(m, \mathcal{D})$.
  $m_2 \leftarrow$ `adjust`$(m_1, \mathcal{D})$.
  $m_3 \leftarrow$ `simplify`$(m_2, \mathcal{D})$.
  If $BIC(m_3|\mathcal{D}) \leq BIC(m|\mathcal{D})$, return m;
  Else $m \leftarrow m_3$.

`expand`$(m, \mathcal{D})$
Repeat until termination:
  $m_1 \leftarrow \arg\max_{m' \in NI(m) \cup SI(m)} IR(m', m|\mathcal{D})$.
  If $BIC(m_1|\mathcal{D}) \leq BIC(m|\mathcal{D})$, return m.
  If $m_1 \in SI(m)$, $m \leftarrow m_1$;
  Else $m \leftarrow$ `localAdjust`$(m_1, m, \mathcal{D})$

`adjust`$(m, \mathcal{D})$
Repeat until termination:
  $m_1 \leftarrow \arg\max_{m' \in NR(m)} BIC(m'|\mathcal{D})$.
  If $BIC(m_1|\mathcal{D}) \leq BIC(m|\mathcal{D})$, return m;
  Else $m \leftarrow m_1$.

`simplify`$(m, \mathcal{D})$
Repeat until termination:
  $m_1 \leftarrow \arg\max_{m' \in SD(m)} BIC(m'|\mathcal{D})$.
  If $BIC(m_1|\mathcal{D}) \leq BIC(m|\mathcal{D})$, break;
  Else $m \leftarrow m_1$.
Repeat until termination:
  $m_1 \leftarrow \arg\max_{m' \in ND(m)} BIC(m'|\mathcal{D})$.
  If $BIC(m_1|\mathcal{D}) \leq BIC(m|\mathcal{D})$, return m;
  Else $m \leftarrow m_1$.

Figure 3: The EAST search procedure.

Conceptually EAST works with unrooted LT models. In implementation, however, we represent unrooted models as rooted models. Rooted LT models are BNs and their parameters are clearly defined. This makes it easy to see how the parameter composition of a candidate model m' is related to that of the current model m. Consider the models m and m' in Figure 1 (a) and (c). m' is obtained from m by introducing a new latent variable Y_4 to mediate Y_3 and two of its neighbors X_6 and X_7. The two models share the parameters for describing the distributions $P(Y_1)$, $P(Y_2|Y_1)$, $P(X_1|Y_2)$, $P(X_2|Y_2)$, $P(X_3|Y_2)$, $P(X_4|Y_1)$, $P(Y_3|Y_1)$ and $P(X_5|Y_3)$. On the other hand, the parameters for describing $P(Y_4|Y_3)$, $P(X_6|Y_4)$ and $P(X_7|Y_4)$ are peculiar to m' while those for describing $P(X_6|Y_3)$ and $P(X_7|Y_3)$ are peculiar to m.

We write the parameters of a candidate model m' as a pair (δ_1, δ_2), where δ_1 is the collection of parameters that m' shares with m. Similarly, we write the parameters of the current model m as a pair (θ_1, θ_2), where θ_1 is the collection of parameters that m shares with m'. Suppose we have computed MLE (θ_1^*, θ_2^*) of the parameters of m. For a given value of δ_2, $(m', \theta_1^*, \delta_2)$ is a fully specified BN. In this BN, we can compute

$$P(\mathcal{D}|m', \theta_1^*, \delta_2) = \prod_{\mathbf{d} \in \mathcal{D}} P(\mathbf{d}|m', \theta_1^*, \delta_2).$$

As a function of δ_2, this will be referred to as the *restricted likelihood function* of δ_2. The *maximum restricted loglikelihood*, or simply the *maximum RL*, of the candidate model m' is defined to be

$$\max_{\delta_2} \log P(\mathcal{D}|m', \theta_1^*, \delta_2).$$

The *restricted likelihood (RL)* method for model evaluation replaces the likelihood term in the BIC score of m' with its maximum RL and uses the resulting approximate score to evaluate m'.

Local EM is a procedure for computing the maximum RL of a candidate model m'. It works in the same way as EM except with the value of δ_1 fixed at θ_1^*. It starts with an initial value $\delta_2^{(0)}$ for δ_2 and iterates. After $t-1$ iterations, it obtains $\delta_2^{(t-1)}$. At iteration t, it completes the data $\mathcal{D}$ using the BN $(m', \theta_1^*, \delta_2^{(t-1)})$, calculates some sufficient statistics, and therefrom obtains $\delta_2^{(t)}$. Suppose the parameters δ_2 of m' describe distributions $P(Z_j|W_j)$ $(j = 1, \ldots, \rho)$. The distributions $P(Z_j|W_j, \delta_2^{(t)})$ that make up $\delta_2^{(t)}$ can be obtained in two steps:

- E-Step: For each data case $\mathbf{d} \in \mathcal{D}$, make inference in the BN $(m', \theta_1^*, \delta_2^{(t-1)})$ to compute $P(Z_j, W_j|\mathbf{d}, m', \theta_1^*, \delta_2^{(t-1)})$.
- M-Step: Obtain
  $P(Z_j|W_j, \delta_2^{(t)}) = f(Z_j, W_j)/\sum_{Z_j} f(Z_j, W_j)$,
  where the *sufficient statistic*
  $f(Z_j, W_j) = \sum_{\mathbf{d} \in \mathcal{D}} P(Z_j, W_j|\mathbf{d}, m', \theta_1^*, \delta_2^{(t-1)})$.

Local EM converges. The series of loglikelihood $\log P(\mathcal{D}|m', \theta_1^*, \delta_2^{(t)})$ increase monotonically with t. This fact can be seen if one examines the proof for the same result about EM. Like EM, local EM might converge to local maxima.

When implemented properly, local EM is much more efficient than EM. There are two reasons. First, fewer sufficient statistics are computed in local EM than in EM. EM requires sufficient statistics for each pair of neighboring variables, while local EM needs the sufficient statistics only for those pairs that are affected by the search operation. Second,

there are abundant opportunities for computation sharing. Consider calculating the posterior $P(Z_j, W_j|\mathbf{d}, m', \theta_1^*, \delta_2^{(t-1)})$ in a candidate model $(m', \theta_1^*, \delta_2^{(t-1)})$. The candidate model is the same as the current model $(m, \theta_1^*, \theta_2^*)$ except for the part affected by the search operation. Consequently, some of the computational steps can be pre-computed in the current model, and they can be shared at all iterations and among different candidate models.

Local maxima is an issue in local EM as well as in EM. To avoid local maxima, we adopt the *pyramid scheme* proposed by Chickering and Heckerman (1997). The idea is to randomly generate μ initial values for the new parameters δ_2, resulting in μ initial models. One local EM iteration is run on all the models and afterwards the bottom $\mu/2$ models are discarded. Then two local EM iterations are run on the remaining models and afterwards the bottom $\mu/4$ models were discarded. Then four local EM iterations are run on the remaining models and so on. The process continues until there is only one model. After that some more local EM iterations are run on the only remaining model to refine the parameters until the total number of iterations reaches a predetermined number ν.

We have described the RL method for implementing the arg max operators in EAST. It takes a list of candidate models as input and outputs a single model. Full EM is run on the output model to optimize its parameters before the search moves on to the next step.

4 Model Evaluation based on Data Completion

We now turn to the *data completion (DC)* method for efficient model evaluation. The method first completes the data set $\mathcal{D}$ using the current model (m, θ^*), where θ^* is the MLE of the parameters of m. It then uses the completed data set $\bar{\mathcal{D}}$ to evaluate candidate models. The completed data set is used in different ways in different subroutines of EAST.

A candidate model m' in the `adjust` subroutine shares the same variables as m. All the variables are observed with respect to $\bar{\mathcal{D}}$. Hence it is easy to compute the maximum expected loglikelihood $\max_{\theta'} \log P(\bar{\mathcal{D}}|m', \theta')$. The DC method replaces the first term of the BIC score by the maximum expected loglikelihood and uses the resulting function to evaluate m'. This idea is due to Friedman (1997) and it is also used in the `localAdjust` subroutine.

Next consider a candidate model m' in the second loop in the `simplify` subroutine. Suppose it is obtained from m by deleting a latent node Z. Let $\bar{\mathcal{D}}_Z$ be the data set obtained from $\bar{\mathcal{D}}$ by removing the values for Z. Then all the variables in m' are observed with respect to $\bar{\mathcal{D}}_Z$. The DC method replaces the first term of the BIC score by $\max_{\theta'} \log P(\bar{\mathcal{D}}_Z|m', \theta')$ and uses the resulting function to evaluate m'. This idea is from Zhang and Kočka (2004).

The SD operator deletes a state from the domain of a latent variable. A related operator is *state merging*. It merges two states of a latent variable. Suppose a candidate model m' is obtained from m by merging two states, i and j, of a latent variable Z. Use $\hat{ij}$ to denote the merged state. Let $\bar{\mathcal{D}}_{\hat{ij}}$ be the data set obtained from $\bar{\mathcal{D}}$ by replacing both i and j with $\hat{ij}$. Then all the variables in m' are observed with respect to $\bar{\mathcal{D}}_{\hat{ij}}$. One can replace the first term of the BIC score by $\max_{\theta'} \log P(\bar{\mathcal{D}}_{\hat{ij}}|m', \theta')$ and use the resulting function to evaluate m'. This idea is borrowed from Elidan and Friedman (2005).

Now suppose m' is a candidate model in the first loop in the `simplify` subroutine, obtained from m by reducing the cardinality of a latent variable Z by 1. The DC method considers all possible ways to achieve the cardinality reduction through state merging, obtains a model score for each way using the above method, and uses the maximum of the scores to evaluate m'.

The candidate models in the `expand` subroutine are generated by two different operators. The DC method first separately evaluates candidate models generated by different operators, picks one model for each operator, runs full EM on the two resultant models and selects between them using improvement ratios.

Let m' be a candidate model obtained from the current model m by introducing a latent node Z to mediate a latent node Y and two

of its neighbors Z_1 and Z_2. Intuitively the new node would be necessary if, in the current model m, Z_1 and Z_2 are not conditionally independent given Y. This condition can be tested based on $\bar{\mathcal{D}}$ because all the three variables are observed in $\bar{\mathcal{D}}$. So the DC method evaluates m' using the G-squared statistic, computed from $\bar{\mathcal{D}}$ for testing the hypothesis that Z_1 and Z_2 are conditionally independent given Y. The larger the statistic, the further away Z_1 and Z_2 are from being independent given Y, and hence the more necessary it is to introduce the new node Z. This idea is due to Zhang and Kočka (2004).

Finally consider a candidate model m' obtained from the current model m by adding a new state to the domain of a latent variable Y. Let Z_1, Z_2, ..., Z_k be the neighbors of Y in m. Intuitively the new state would be necessary if the interactions among Z_1, Z_2, ..., Z_k are not properly modeled by Y. Let $\hat{P}$ be the empirical joint distribution computed from $\bar{\mathcal{D}}$ and P_m be the joint distribution given by the model m. For any two neighbors Z_i and Z_j of Y, the KL distance $KL(\hat{P}(Z_i, Z_j) \| P_m(Z_i, Z_j))$ is a measure of how well the interactions between them are modeled. The larger the KL distance, the poorer the interactions are modeled, and hence the more necessary it is to add a new state to Y. The DC method evaluates m' using the KL distance averaged over all pairs of neighbors of Y. This idea is due to Zhang and Kočka (2004).

Note that although the DC method conceptually starts with data completion, it does not compute an explicit representation of $\bar{\mathcal{D}}$. All the heuristic model scores can be computed from the current model (m, θ^*) and the original data set $\mathcal{D}$ and the computations all boil down to calculating sufficient statistics on some variables. Full EM is run on the model picked by any arg max operator to optimize its parameters before the search moves on to the next step.

5 Empirical Results

Coupling the EAST search procedure with either the RL method or the DC method for model evaluation, one obtains two different algorithms for learning LT models, which we denote by *EAST-RL* and *EAST-DC* respectively. The main purpose of our empirical studies is to compare the two algorithms.

We also attempt to answer two other questions. First, EAST-RL has two parameters μ and ν. How do they influence the performance of the algorithm? Second, the cost-effectiveness principle mentioned in section 2 was introduced to deal with the problem of operation granularity, i.e. some operations might increase model complexity much more than others. Is the principle necessary given that model complexity is considered already in the BIC score?

The Data: The synthetic data used in our experiments were generated using three manually constructed LT models that contain 7, 12 and 18 manifest variables respectively. Three data sets of sizes 1k, 5k and 10k were sampled from the 18 variable model. We denote them by $\mathcal{D}_{18}(1k)$, $\mathcal{D}_{18}(5k)$ and $\mathcal{D}_{18}(10k)$. One data set was sampled from each of the other two models. They consists of 5k and 10k samples and hence are denoted by $\mathcal{D}_7(5k)$ and $\mathcal{D}_{12}(10k)$. The data sets were analyzed by various algorithms and their variants. The quality of a learned model is measured by the empirical KL distance of the model to the corresponding generative model, an approximation to the true KL distance that was computed based on 5k testing data. The results are shown in Table 1. They are averaged over 10 runs. The standard deviations are given in the parentheses.

There are three real-world data sets. Their basic information is given in the table.

	# vars	# states per var	sample size train	sample size test
ICAC	31	3.5	1200	301
KIDNEY	35	4.0	2000	600
COIL	42	2.7	5822	4000

The COIL data set originates from the COIL Challenge 2000 (van der Putten and van Someren 2004). It consists of customer records of a Dutch insurance company. The ICAC data set is from a telephone survey by Hong Kong's anti-corruption agency on public perception about various issues related to corruption. KIDNEY is a medical data set studied by Zhang *et al.* (2008). Each of the data sets was split into two subsets, one for training and

Table 1: Empirical results on synthetic data (the top two) and on real-world data (the bottom).

	$\mathcal{D}_7(5k)$		$\mathcal{D}_{12}(10k)$	
	emp-KL	time(mins)	emp-KL	time(hrs)
EAST-RL(μ,ν) (1,20)	0.0117(2.1e-3)	3.8(2.2)	0.0062(4.2e-3)	1.8(0.3)
(8,20)	0.0105(1.3e-3)	5.9(1.6)	0.0049(3.8e-3)	2.2(0.3)
(8,40)	0.0101(4.5e-5)	7.1(0.1)	0.0032(2.4e-4)	2.6(0.2)
EAST-DC	0.0101(5.7e-5)	5.8(0.1)	0.0051(5.0e-3)	1.4(0.1)
EAST-RL0(μ, ν) (8,40)	0.0101(4.8e-05)	6.3(0.1)	0.0079(4.7e-3)	1.5(0.1)

	$\mathcal{D}_{18}(1k)$		$\mathcal{D}_{18}(5k)$		$\mathcal{D}_{18}(10k)$	
	emp-KL	time(hrs)	emp-KL	time(hrs)	emp-KL	time(hrs)
EAST-RL(μ,ν) (1,20)	0.1865(1.5e-5)	0.5(0.03)	0.0245(9.1e-3)	4.3(0.7)	0.0097(4.0e-3)	9.4(1.1)
(8,20)	0.1865(2.3e-5)	0.6(0.05)	0.0175(5.3e-3)	5.7(0.9)	0.0067(2.5e-3)	13.8(1.6)
(8,40)	0.1865(7.5e-6)	0.7(0.02)	0.0148(4.5e-3)	6.0(0.6)	0.0047(7.0e-4)	18.4(3.9)
EAST-DC	0.2171(3.3e-2)	0.6(0.04)	0.0371(3.5e-3)	3.9(0.4)	0.0113(3.0e-3)	8.2(1.5)
EAST-RL0(μ, ν) (8,40)	0.1865(6.3e-06)	0.6(0.01)	0.0326(1.1e-2)	4.4(0.9)	0.0207(1.2e-2)	10.1(1.8)

	KIDNEY			COIL			ICAC		
EAST-RL	BIC	logscore	time(days)	BIC	logscore	time(days)	BIC	logscore	time(days)
(μ,ν)=(4,10)	-57214(61)	-16882(41)	0.4(0.1)	-52116(205)	-34943(203)	0.9(0.2)	-26102(52)	-6198(23)	0.10(0.01)
(8,20)	-57158(73)	-16818(56)	0.6(0.1)	-51773(159)	-34577(195)	1.1(0.2)	-26033(22)	-6167(15)	0.16(0.03)
(16,40)	-57066(52)	-16761(25)	1.0(0.1)	-51505(74)	-34198(41)	2.3(0.4)	-26042(27)	-6173(15)	0.22(0.02)
EAST-DC	-57699(158)	-17236(156)	0.3(0.0)	-52560(295)	-35103(226)	0.7(0.1)	-26156(1)	-6213(13)	0.09(0.00)

the other for testing. LT models were obtained from the training sets. For each learned model, we computed its BIC score on the training set and its logarithmic score on testing set. The logscore measures how well the model predicts future data. The results are shown in Table 1. They are averaged over 5 runs.

Impact of μ and ν: The parameter μ controls the effort that local EM spends on avoiding local maxima, while ν controls the effort that local EM spends on refining parameters. In general we expect EAST-RL to find better models as they increase. Consider the KL distance from a learned model to the corresponding generative model as the parameter setting (μ, ν) changes from (1, 20) to (8, 20) and then to (8, 40). We see that the KL distance changes, on average, from 0.0097 to 0.0067 and then to 0.0047 for $\mathcal{D}_{18}(10k)$; from 0.0245 to 0.0175 and then to 0.0148 for $\mathcal{D}_{18}(5k)$; from 0.0062 to 0.0049 and then to 0.0032 for $\mathcal{D}_{12}(10k)$; and from 0.0117 to 0.0105 and then to 0.0101 for $\mathcal{D}_7(5k)$. It stays unchanged for $\mathcal{D}_{18}(1k)$.

The results on real-world data show similar trends with one exception. In the case of the ICAC data, the BIC score and the logscore drop slightly from (8, 20) to (16, 40), probably due to randomness.

EAST-RL vs. EAST-DC: The main purpose of our empirical studies is to compare EAST-RL and EAST-DC. Consider the experiments with synthetic data first. We compare the models found by the two algorithms in terms of the KL distances from those models to the corresponding generative models. For $\mathcal{D}_{18}(10k)$, the average KL distance of the models found by EAST-DC is 0.0113, while that of the models found by EAST-RL is 0.0097 in the lowest parameter setting and 0.0047 in the highest setting. As a matter of fact, EAST-RL found better models for $\mathcal{D}_{18}(10k)$ than EAST-DC in all the parameter settings considered. The same is true for $\mathcal{D}_{18}(5k)$ and $\mathcal{D}_{18}(1k)$. For $\mathcal{D}_{12}(10k)$, EAST-RL found better models than EAST-DC in all the settings except for (1, 20). For $\mathcal{D}_7(5k)$, EAST-RL found models of the same quality as EAST-DC in the highest setting.

EAST-RL also performed better than EAST-DC on the real-world data. It found better models on all the data sets and in all the settings.

Running times are also reported in Tables 1. They were collected on an Intel(R) Core(TM)2 PC with clock rate of 2.4GHz. EAST-DC is clearly more efficient than EAST-RL. On the COIL data, for instance, it was about 4 times faster than EAST-RL in the setting (16, 40).

Operator Granularity: In Table 1, EAST-RL0 denotes an implementation of EAST-RL where operation granularity is not considered when evaluating candidate models generated by SI and NI. We see that the models that EAST-RL0 found for $\mathcal{D}_{18}(10k)$, $\mathcal{D}_{18}(5k)$ and $\mathcal{D}_{12}(10k)$ are significantly worse than those found by EAST-RL. They are tied on the other

cases. This suggests that it is necessary to deal with operation granularity using the cost-effectiveness principle in the `expand` subroutine.

6 Discussions and Conclusions

This paper is concerned with the search-based approach to learning latent tree model. A key problem in the approach is how to efficiently evaluate large numbers of candidate models. A variety of ideas for attacking the problem were previously proposed by Zhang and Kočka (2004) and Elidan and Friedman (2005). In this paper we observe that those ideas can be grouped into two distinct approaches, namely the restricted likelihood (RL) approach and the data completion (DC) approach. We study and compare the approaches in the framework of EAST, a newly developed search procedure for learning latent tree models. This is the first time that the two approaches to efficient model evaluation are clearly identified and studied.

The RL method is conceptually simpler than the DC method. It is based on one principle, while the DC method is based on several heuristic ideas. The RL method is easier to understand than the DC method.

Ideally one should select the candidate model with the maximum BIC score. In the RL method, we replace the likelihood term in the BIC score with the maximum restricted loglikelihood. What it results in is an approximation that lower bounds the BIC score. So the RL method selects a model to maximize a lower bound of the true objective function. This is common practice in machine learning.

The DC method also selects candidate models to maximize some objective functions. However it is less clear how those functions are related to the BIC score. In the case of the `adjust` subroutine, some relationship exists because $\max_{\theta'} \log P(\bar{\mathcal{D}}|m', \theta') \geq \max_{\theta} \log P(\bar{\mathcal{D}}|m, \theta)$ implies $\max_{\theta'} \log P(\mathcal{D}|m', \theta') \geq \max_{\theta} \log P(\mathcal{D}|m, \theta)$. The same relationship is not known to be true for the other cases.

We empirically tested EAST-RL and EAST-DC on a number of synthetic and real-world data sets. Several parameter settings were tried for EAST-RL. At the lowest setting, EAST-RL found better models than EAST-DC on almost all the data sets and it took roughly the same amounts of time on almost all the data sets. At the highest setting, EAST-RL found better models than EAST-DC on all the data sets and much better models on many of the data sets. However, it was also significantly slower.

Acknowledgements

We thank Kin Man Poon for valuable discussions. Research on this work was supported by Hong Kong Grants Council Grants #622307, and China National Basic Research Program (aka the 973 Program) under project No.2003CB517106. The work was completed when the third author was on leave at the HKUST Fok Ying Tung Graduate School.

References

D. M. Chickering (2002). Learning Equivalence Classes of Bayesian-Network Structures. *JMLR*, 2.

D. M. Chickering and D. Heckerman (1997). Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29.

R. Durbin, S. Eddy, A. Krogh, and G. Mitchison (1998) *Biological sequence analysis: probabilistic models of proteins and nucleic acids.* Cambridge Univ. Press.

G. Elidan and N. Friedman (2005). Learning hidden variable networks: the information bottleneck approach. *Journal of Machine Learning Research*, 6:81-127.

N. Friedman (1997). Learning belief networks in the presence of missing values and hidden variables. *ICML*.

D. Geiger, D. Heckerman and C. Meek (1996). Asymptotic Model Selection for Directed Networks with Hidden Variables. *UAI-96*, 158-168.

S. Guindon and O. Gascuel (2003). A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Systematic Biology*, 52(5):696-704.

P. F. Lazarsfeld and N. W. Henry (1968). *Latent structure analysis.* Houghton Mifflin, Boston.

I. Nachman, G. Elidan and N. Friedman (2004). "Ideal Parent" Structure Learning for Continuous Variable Networks. *UAI-04*, 400-409.

J. Pearl (1988). *Probabilistic reasoning in intelligence systems.* Morgan Kaufmann, San Mateo.

P. van der Putten and M. van Someren. A bias-variance analysis of a real world learning problem: The COIL challenge 2000. *Machine Learning*, 57:177-195.

N. L. Zhang (2004). Hierarchical latent class models for cluster analysis. *JMLR*, 5:697-723.

N. L. Zhang (2007). Discovery of Latent Structures: Experience with the CoIL Challenge 2000 Data Set. *ICCS-2007*, 26-34.

N. L. Zhang and T. Kočka (2004). Efficient learning of hierarchical latent class models. *ICTAI-2004*, 585-593.

N. L. Zhang, S. H. Yuan, T. Chen and Y. Wang (2008). Latent tree models and diagnosis in traditional Chinese medicine. *AI in Medicine*, 42:229-245.