Exploiting Qualitative Domain Knowledge for Learning Bayesian Network Parameters with Incomplete Data

Wenhui Liao
Thomson-Reuters Corporation
wenhui.liao@thomsonreuters.com

Qiang Ji
Rensselaer Polytechnic Institute
qji@ecse.rpi.edu

Abstract

When a large amount of data are missing, or when multiple hidden nodes exist, learning parameters in Bayesian networks (BNs) becomes extremely difficult. This paper presents a learning algorithm to incorporate qualitative domain knowledge to regularize the otherwise ill-posed problem, limit the search space, and avoid local optima. Specifically, the problem is formulated as a constrained optimization problem, where an objective function is defined as a combination of the likelihood function and penalty functions constructed from the qualitative domain knowledge. Then, a gradient-descent procedure is systematically integrated with the E-step and M-step of the EM algorithm, to estimate the parameters iteratively until it converges. The experiments show our algorithm improves the accuracy of the learned BN parameters significantly over the conventional EM algorithm.

1 Introduction

Many real applications need to automatically learn BN parameters from data due to the difficulty and time requirement of manually setting up the parameters. In reality, training data are usually incomplete, or complete but spares because some events rarely happen. Various approaches have been proposed to learn parameters when data is missing. The classical approaches include the Expectation-Maximization (EM) algorithm [1] and Gibbs Sampling [5]. Other methods are proposed to overcome the disadvantages of EM and Gibbs sampling, including the AI&M procedure [6], the RBE algorithm [9], the Information-bottleneck EM algorithm [3], data perturbation method [4], and others.

The above methods deal with different aspects of parameter learning, however, when data are missing completely at random, in other words, when several hidden (unobserved) nodes exist, the learned parameters could be quite different from the true parameters. Furthermore, most methods do not emphasize domain knowledge or lack a systematic strategy to incorporate the domain knowledge into the machine learning methods. In most domains, at least some information, either from literature or from domain experts, is available about the model to be constructed. However, many forms of prior knowledge that an expert might have are difficult to be directly used by existing machine learning algorithms. Therefore, it is important to formalize the knowledge clearly and incorporate it into learning.

This motivates us to propose a BN learning algorithm for the case when multiple hidden nodes exist by systematically combining domain knowledge during learning. Instead of using quantitative domain knowledge, which is often hard to obtain, we propose to exploit qualitative domain knowledge. Qualitative domain knowledge impose approximated constraints on some parameters or on the relationships among some parameters. These kinds of qualitative knowledge are often readily available. Specifically, two qualitative constraints are considered, the range of parameters, and the relative relationships between different parameters. Instead of using the likelihood function as the objective to maximize during learning, we define the objective function as a combination of the likelihood function and the penalty functions constructed from the constraints. Then, a gradient-descent procedure is systematically integrated with the E-step and M-step of the EM algorithm, to estimate the parameters iteratively until it converges. The experiments show the proposed algorithm is promising to improve the accuracy of the learned BN parameters over the EM method.

2 Learning BN Parameters

2.1 Qualitative Constraints with Confidence

Let G be a BN with nodes $X_1, ..., X_n$. We use θ to denote the entire vector of parameter value θ_{ijk}, $\theta_{ijk} =$

$p(x_i^k|pa_i^j)$, where i $(i = 1, ..., n)$ ranges over all the variables in the BN, j $(j = 1, ..., q_i)$ ranges over all the possible parent configurations of X_i, and k $(k = 1, ..., r_i)$ ranges over all the possible states of X_i. Given the data set D, the goal of parameter learning is to find the most probable values $\hat{\theta}$ for θ that can maximize the log likelihood L_D.

In many real-world applications, domain experts usually have valuable information about model parameters θ_{ijk}. We consider two types of constraints: type-I is about the range of a parameter; and type-II is about the relative relationships $(>, <, =)$ between different parameters. One of our goals is to make the constraints as simple as possible, so that the experts can easily formalize their knowledge into these constraints.

The range of a parameter allows domain experts to specify an upper bound and a lower bound for the parameter, instead of defining the specific values. In addition to assessing the ranges of parameters, the domain experts may also know the relative relationships between some parameters. For each type-II constraint, if the two associated parameters are in the same CPT of a node, we call it an inner-relationship constraint; if the two parameters come from CPTs of different nodes, we call it an outer-relationship constraint.

Let A be the set that includes the parameters whose ranges are known based on the domain knowledge. For each $\theta_{ijk} \in A$, we define the range as $l_{ijk} \leq \theta_{ijk} \leq u_{ijk}$. Obviously, $l_{ijk} \geq 0$, and $u_{ijk} \leq 1$. Let B be the set that includes the parameters whose relative relationships are known based on the domain knowledge. For each $\theta_{ijk} \in B$, we can find another $\theta_{i'j'k'}$ so that $\theta_{ijk} \geq \theta_{i'j'k'}$, or/and, $\theta_{ijk} = \theta_{i'j'k'}$, where $i \neq i'$, or $j \neq j'$, or $k \neq k'$. However, the domain knowledge may not be reliable all the time. We thus associate confidence levels λ_{ijk}, $\lambda_{ijk}^{i'j'k'}$ to each constraint in the sets A and B respectively. The value of each λ is between 0 and 1 (1 indicates high confidence).

2.2 Parameter Learning with Qualitative Constraints

Our goal is to find the optimal parameter $\hat{\theta}$ that maximizes log likelihood $L_D(\theta)$ (D is training data) given the three constraints as below:

$$\begin{array}{ll} \text{Maximize} & L_D(\theta) \\ \text{Subject to} & \sum_k \theta_{ijk} = 1, \\ & 1 \leq i \leq n, 1 \leq j \leq q_i, 1 \leq k \leq q_i \\ & l_{ijk} \leq \theta_{ijk} \leq u_{ijk}, \theta_{ijk} \in A \\ & \theta_{ijk} \geq \theta_{i'j'k'}, \theta_{ijk}, \theta_{i'j'k'} \in B \end{array} \quad (1)$$

For each inequality constraint, we define the following penalty functions:

$$g'(\theta_{ijk}) = [\theta_{ijk} - l_{ijk}]^-, \forall \theta_{ijk} \in A \quad (2)$$

$$g''(\theta_{ijk}) = [u_{ijk} - \theta_{ijk}]^-, \forall \theta_{ijk} \in A \quad (3)$$

$$g'''(\theta_{ijk}, \theta_{i'j'k'}) = [\theta_{ijk} - \theta_{i'j'k'}]^-, \forall \theta_{ijk}, \theta_{i'j'k'} \in B \quad (4)$$

where $[x]^- = \max(0, -x)$.

Therefore, we can rephrase Equation 1 as follows:

$$\begin{array}{l} \text{Maximize} \\ J(\theta) = L_D(\theta) - \frac{w_1}{2} \sum_{\theta_{ijk} \in A} \lambda_{ijk} [(g'(\theta_{ijk}))^2 + (g''(\theta_{ijk}))^2] \\ - \frac{w_2}{2} \sum_{\theta_{ijk} \in B} \lambda_{ijk}^{i'j'k'} (g'''(\theta_{ijk}, \theta_{i'j'k'}))^2 \\ \text{Subject to} \sum_k \theta_{ijk} = 1 \end{array} \quad (5)$$

where w_i is the penalty weight, which is decided empirically. Obviously, the penalty varies with the confidence level for each constraint.

In order to solve the constrained optimization problem, first, we eliminate the constraint $\sum_k \theta_{ijk} = 1$ by reparameterizing θ_{ijk} so that the new parameters automatically respect the constraint on θ_{ijk} no matter what their values are. We define a new parameter β_{ijk} such that

$$\theta_{ijk} \equiv \frac{\exp(\beta_{ijk})}{\sum_{k'=1}^{r_i} \exp(\beta_{ijk'})} \quad (6)$$

In this way, a local maximum w.r.t. to β_{ijk} is also a local maximum w.r.t. θ_{ijk}, and vice versa. Most importantly, the constraint is automatically satisfied.

Next is to get the derivative of $J(\theta)$ w.r.t. β. $\nabla_{\theta_{ijk}} L_D(\theta)$ can be expressed below [7]:

$$\nabla_{\theta_{ijk}} L_D(\theta) = \frac{\sum_{l=1}^N p(x_i^k, pa_i^j|D_l, \theta)}{\theta_{ijk}} \quad (7)$$

Therefore, based on the chain rule,

$$\begin{aligned} \nabla_{\beta_{ijk}} L_D(\theta) &= \frac{\partial L_D(\theta)}{\partial \theta_{ijk}} \frac{\partial \theta_{ijk}}{\partial \beta_{ijk}} \\ &= \nabla_{\theta_{ijk}} L_D(\theta)(\theta_{ijk} - \theta_{ijk}^2) \\ &= \sum_{l=1}^N p(x_i^k, pa_i^j|D_l, \theta)(1 - \theta_{ijk}) \end{aligned} \quad (8)$$

Similarly, for $g'(\theta_{ijk})$, $g''(\theta_{ijk})$, and $g'''(\theta_{ijk})$, the derivatives are:

$$\nabla_{\beta_{ijk}} g'(\theta_{ijk}) = \begin{cases} \theta_{ijk}^2 - \theta_{ijk}; & \text{if } \theta_{ijk} \leq l_{ijk} \\ 0; & \text{otherwise} \end{cases} \quad (9)$$

$$\nabla_{\beta_{ijk}} g''(\theta_{ijk}) = \begin{cases} \theta_{ijk} - \theta_{ijk}^2; & \text{if } \theta_{ijk} \geq u_{ijk} \\ 0; & \text{otherwise} \end{cases} \tag{10}$$

$$\nabla_{\beta_{ijk}} g'''(\theta_{ijk}, \theta_{i'j'k'}) = \begin{cases} \theta_{ijk}^2 - \theta_{ijk}; \\ \text{if } \theta_{ijk} < \theta_{i'j'k'},\ i \neq i', \text{or } j \neq j' \\ \theta_{ijk}^2 - \theta_{ijk} - \theta_{ijk}\theta_{i'j'k'}; \\ \text{if } \theta_{ijk} < \theta_{i'j'k'},\ i = i',\ j = j' \\ 0;\ \text{otherwise} \end{cases} \tag{11}$$

Therefore, the derivative of $J(\theta_{ijk})$ w.r.t. β_{ijk} is:

$$\begin{aligned} &\nabla_{\beta_{ijk}} J(\theta_{ijk}) = \textstyle\sum_{l=1}^{N} P(x_i^k, pa_i^j | D_l, \theta)(1 - \theta_{ijk}) \\ &-w_1 \lambda_{ijk} [g'(\theta_{ijk}) \nabla_{\beta_{ijk}} g'(\theta_{ijk}) + g''(\theta_{ijk}) \nabla_{\beta_{ijk}} g''(\theta_{ijk})] \\ &-w_2 \sum_{B^+(\theta_{ijk})} [\lambda_{ijk}^{i'j'k'} g'''(\theta_{ijk}, \theta_{i'j'k'}) \nabla_{\beta_{ijk}} g'''(\theta_{ijk}, \theta_{i'j'k'})] \\ &+w_2 \sum_{B^-(\theta_{ijk})} [\lambda_{ijk}^{i'j'k'} g'''(\theta_{i'j'k'}, \theta_{ijk}) \nabla_{\beta_{ijk}} g'''(\theta_{i'j'k'}, \theta_{ijk})] \end{aligned} \tag{12}$$

where $B^+(\theta_{ijk})$ is the set of the constraints whose first term is θ_{ijk}, while $B^-(\theta_{ijk})$ is the set of the constraints whose second term is θ_{ijk}. Both $B^+(\theta_{ijk})$ and $B^-(\theta_{ijk})$ belong to the set B.

Table 1. A constrained EM (CEM) learning algorithm

Repeat until it converges
- Step 1: E-Step to compute the conditional expectation of the log-likelihood function;
- Step 2: M-step to find the parameter θ' that maximizes the expected log-likelihood;
- Step 3: Perform the following optimization procedure based on the gradient descent method:
  - $\theta^t = \theta'$; map θ^t to β^t based on Equation 6
  - Repeat until $\Delta\beta^t \simeq 0$
    - $\Delta\theta^t = 0$
    - for each variable i, parent configuration j, value k
      - for each $D_l \in D$
        - $\Delta\theta_{ijk}^{t+1} = \Delta\theta_{ijk}^t + p(x_i^k, pa_i^j | D_l, \theta_t)$
      - $\Delta\beta_{ijk}^{t+1} = \Delta\theta_{ijk}^{t+1}(1 - \theta_{ijk}) + K$
      - (K represents the last three terms in Eq. 12)
    - $\beta^{t+1} = \beta^t + \Delta\beta^{t+1}$
  - map β^{t+1} to θ^{t+1} based on Equation 6
  - $\theta^t = \theta^{t+1}$
- Go to Step 1

Return θ^t

Now, we are ready to present the constrained EM (CEM) learning algorithm as summarized in Table 1. The algorithm consists of three steps. The first two steps are the same as the E-step and M-step in the EM algorithm. However, in the third step, a gradient-based update is used to force the solutions to move towards the direction of reducing constraint violations.

3 An Illustrative Application

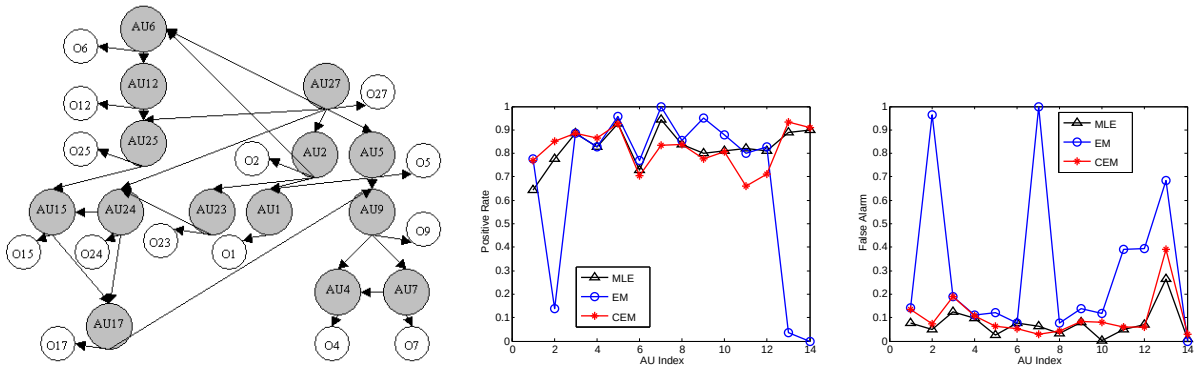
In recent years, a variety of approaches are proposed to recognize facial expressions. Instead of recognizing only six basic facial expressions directly, people also developed techniques to automatically recognize facial action units (AUs) [2], which could compose a rich collection of facial expressions. Since different AUs are closely related with each other, instead of recognizing each AU alone, a BN can be used to model the probabilistic relationships among different AUs. Figure 1(a) illustrates such a BN to model 14 AUs.

In the model, each AU is connected with a measurement node (unshaded node), which encodes the measurement obtained from Adaboost classifers[10]. To learn the model parameters, complete training samples consisting of the true AU labels and the obtained measurements are desired. However, manually labeling AUs is usually time consuming and difficult. Sometimes, the labeled data could be wrong due to human subjective. In addition, some AU events rarely happen in the collected data. Therefore, the training data could be incomplete, biased, or spare for certain events. We thus apply our algorithm to learn the parameters.

We first generate constraints. For each AU node, the domain experts are able to figure out the ranges for most of the parameters. For each measurement node, since the performance of each Adaboost classifier for each AU varies, we can rank their performance and transfer such a ranking into the outer-relationships between these measurement nodes. For example, the Adaboost classifier performs better recognition of AU2 (outer brow raiser) than AU23 (lip tighten), then we can get constraints like $p(O2 = 0|AU2 = 0) > p(O23 = 0|AU23 = 0)$, $p(O2 = 1|AU2 = 1) > p(O23 = 1|AU23 = 1)$, where 0 means an AU is absent, and 1 means an AU is present.

More type-II constraints can be obtained based on the properties of different AUs. For example, for AU6 (cheek raiser) and AU12 (lip corner puller), the probability of AU12 being absent if AU6 is absent, is smaller than the probability of AU6 being present if AU12 is present, $p(AU12 = 0|AU6 = 0) < p(AU12 = 1|AU6 = 1)$. For AU1 (inner brow raiser), the influence of AU2 (outer brow raiser) is larger than the influence of AU5 (upper lid raiser), $p(AU1 = 1|AU2 = 1, AU5 = 0) > p(AU1 = 1|AU2 = 0, AU5 = 1)$.

We use 8000 images collected from Cohan and Kanade's DFAT-504 database [8], where 80% are used for training and 20% data are used for testing. We first use MLE to learn parameters from the complete data, which consists both the true AU labels and the AU measurements. Then, we use the EM and CEM algorithms



(a) (b) (c)

Figure 1. (a) A Bayesian network for AU modeling. We adapted it from [10]. The unshaded nodes are measurement nodes. AU1: inner-brow raiser, AU2: outer-brow raiser, AU4: brow lowerer, AU5: upper-lid raiser, AU6: cheek raiser, AU7: lid tighten, AU9: nose wrinkle, AU12: lip-corner depressor, AU17: chin raiser, AU23:lip tighten, AU24: lip presser, AU25: lips part, AU27: mouth stretch. (b)(c)AU recognition results with the BNs learned from MLE, EM, and CEM respectively: (b) positive rate; (c) false alarm. MLE uses complete data, while EM and CEM use incomplete data that only include AU measurements.

to learn parameters from the incomplete data, which only includes the AU measurements.

Figure 1 compares the AU recognition results with BNs learned from MLE, EM, and CEM in terms of positive rate and false alarm. CEM performs similarly to MLE (based on complete data), but much better than EM. The positive rate increases from 0.69 (EM) to 0.82 (CEM), and the false alarm decreases from 0.32 (EM) to 0.1 (CEM). For EM, some AUs totally fail, such as AU2, AU7, and AU23. But CEM has a fair performance for all the AUs even it is learned from the unlabeled data. This again shows the importance of domain knowledge. Compared to MLE that is based on the labeled data, the CEM has comparable performance but without using any labeled AUs.

4 Conclusion

We present a constrained EM algorithm to learn BN parameters when a large amount of data are missing in the training data. The algorithm fully utilizes qualitative domain knowledge to regularize the otherwise ill-posed problem. Compared with the quantitative domain knowledge such as prior probability distribution typically used by the existing methods, the qualitative domain knowledge is local (only concerned with some parameters), easy to specify, and does not need strong assumption. The experiments show that our algorithm improves the accuracy of the learned parameters significantly over the traditional EM.

5 Acknowledgement

This work is supported in part by a grant from the U.S. Army Research Office under grant number W911NF-06-1-0331.

References

[1] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *The Royal Statistical Society Series B*, 39:1–38, 1977.

[2] P. Ekman and W. Friesen. *Facial Action Coding System: A Technique for the Measurement of Facial Movement*. Consulting Psychologists Press, Palo Alto, CA, 1978.

[3] G. Elidan and N. Friedman. The information bottleneck em algorithm. *UAI*, pages 200–209, 2003.

[4] G. Elidan, M. Ninio, N. Friedman, and D. Schuurmans. Data perturbation for escaping local maxima in learning. *AAAI*, 2002.

[5] S. Geman and D. Geman. Stochastic relaxation, gibbs distribution and the bayesian restoration of images. *PAMI*, 6, 1984.

[6] M. Jaeger. The ai&m procedure for learning from incomplete data. *UAI*, pages 225–232, 2006.

[7] J.Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, pages 213–244, 1997.

[8] T. Kanade, J. F. Cohn, and Y. Tian. Comprehensive database for facial expression analysis. *Proc. of FG00*, 2000.

[9] M. Ramoni and P. Sebastiani. Robust learning with missing data. *Machine Learning*, 45(2):147–170, 2001.

[10] Y. Tong, W. Liao, and Q. Ji. Inerring faical action units with causal relations. *CVPR*, 2006.