# Learning Bayesian Networks with Qualitative Constraints

Yan Tong
GE Global Research Center
Niskayuna, NY12309

Qiang Ji
Rensselaer Polytechnic Institute
Troy, NY12180

## Abstract

*Graphical models such as Bayesian Networks (BNs) are being increasingly applied to various computer vision problems. One bottleneck in using BN is that learning the BN model parameters often requires a large amount of reliable and representative training data, which proves to be difficult to acquire for many computer vision tasks. On the other hand, there is often available qualitative prior knowledge about the model. Such knowledge comes either from domain experts based on their experience or from various physical or geometric constraints that govern the objects we try to model. Unlike the quantitative prior, the qualitative prior is often ignored due to the difficulty of incorporating them into the model learning process.*

*In this paper, we introduce a closed-form solution to systematically combine the limited training data with some generic qualitative knowledge for BN parameter learning. To validate our method, we compare it with the Maximum Likelihood (ML) estimation method under sparse data and with the Expectation Maximization (EM) algorithm under incomplete data respectively. To further demonstrate its applications for computer vision, we apply it to learn a BN model for facial Action Unit (AU) recognition from real image data. The experimental results show that with simple and generic qualitative constraints and using only a small amount of training data, our method can robustly and accurately estimate the BN model parameters.*

## 1. Introduction

Graphical models are becoming increasingly popular in computer vision recently. Among these models, Bayesian networks (BNs) have received significant attention. They have been applied to various computer vision problems including image segmentation [17], object detection [14], target tracking [19], and facial expression understanding [20].

Among all the issues of BNs, parameter learning is one of the main challenges. Parameter learning is to estimate the conditional probability distributions (CPDs) for each node, given the structure of a BN. Many learning techniques rely heavily on training data [13]. Ideally, with sufficient data, it is possible to learn the parameters by standard statistical analysis like maximum likelihood (ML) estimation. In many real-world cases, however, the training data are either incomplete or sparse due to various difficulties with data labeling. Data incompleteness is defined as missing of data for some parameters, while data sparseness means the amount of training data is limited. Both data incompleteness and sparseness occur frequently in computer vision. Cohen *et al*. [4] use a BN to recognize emotions through facial expressions displayed in video sequences. Manually labeling the emotion for each image is both time consuming and error-prone, therefore producing the data lacking both in quality and quantity. Mortensen *et al*. [17] introduce a BN based semi-automatic technique for image segmentation. The parameters are manually constructed, as there are no training data available.

When data are incomplete, Expectation-Maximization (EM) algorithm is often used. Most EM-based methods asume that data are missing at random (MAR), which means the missing values can be estimated by the observed ones in some way. However, when data are missing completely at random (MCAR), *e.g*. data of hidden nodes, the learned parameters could be far from the ground truth.

When data are sparse but complete, ML estimation is often used. To help improve the ML estimation accuracy and to compensate the data sparseness, domain knowledge is often incorporated. Currently, the most popular way is to represent domain knowledge as the prior distribution of the parameters. For discrete BNs, Geiger and Heckerman [12] have proved that Dirichlet Priors are the only possible priors, under certain assumptions. However, it is often difficult for domain experts to specify the full prior probability distribution for all the parameters precisely, especially when the BN has a large number of parameters. Other more general priors include smoothness, sparseness, and locality. These constraints are either too general or primarily used for constraining the model structure.

We propose a framework for parameter learning by combining quantitative data with domain knowledge in the form of qualitative constraints. We want to exploit some domain-specific yet generic qualitative constraints to help regularize the model learning. Two kinds of qualitative constraints are defined: range constraints applied to individual parameters and relationship constraints applied to pairs of parameters. For sparse but complete data, we solve the learning task by reformulating the problem as a constrained ML (CML) problem. For incomplete data, we introduce the constrained EM (CEM) by adding constraints to the M step. In addition, we provide closed form solutions to both CML and CEM.

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

## 2. Related Work

We have already discussed that one of the shortcomings of EM algorithms is that it can easily be trapped in a local maximum when data are MCAR. Till now, there are many different methods to help EM to escape from the local maximum, such as the information-bottleneck EM algorithm [9], data perturbing method [10], and the Markov chain Monte Carlo (MCMC) [7] method. These methods focus on improving the machine learning techniques, but ignoring the useful domain knowledge. In addition, they cannot produce closed form solution (e.g. MCMC method).

Qualitative constraints have been exploited in parameter learning. Wittig *et al*. [21] present an iterative method to integrate qualitative constraints into two learning algorithms, APN [3] and EM, by adding violation functions as a penalty term to the log likelihood function. They show that domain knowledge in the form of constraints can improve learning accuracy. However, this penalty-based method cannot guarantee to find the global maximum. Besides, the weights for the penalty functions often need be manually tuned, depending on applications. Altendorf *et al*. [1] describe a method to incorporate monotonicity constraints into learning algorithm. It assumes that the values of the variables can be totally ordered. Additionally, it uses the penalty functions, which suffers from the same problem as [21]. Feelders and Van der Gaag [11] incorporate some simple inequality constraints in the learning process. They assume that all the variables are binary. The constraints used in the above methods [1, 21, 11] are restrictive, as each constraint has to involve all parameters in a conditional probability table (CPT).

Campos and Cozman [5] formulate the learning problem as a constrained optimization problem. However, they do not provide a specific method to solve the optimization problem. Niculescu *et al*. [18] also solve the learning problem by optimization techniques. They derive the closed form solutions with ML estimation for two kinds of constraints: inequalities between sums of parameters and upper bounds on sum of parameters within a CPT. There are two main limitations of their method: First, they assume one parameter can and only can have one constraint, and there is no overlap between parameters of different constraints. Second, their method cannot handle constraints from different CPTs. We improve their method by deriving the closed form solution for range constraints, which contain both upper bound and lower bound constraints for the same parameters. In addition, the relationship constraints defined in our paper can either be within or between CPTs.

## 3. Problem Definition and Approach

### 3.1. Basic Parameter Learning Theory

We focus on parameter learning in a BN with all discrete nodes, where the structure is known in advance. The notations are defined as follows. Assume a BN with $n$ nodes, $\theta$ is the entire vector of parameters, and $\theta_{ijk}$ denotes one of the parameters. $\theta_{ijk} = p(x_i^k|pa_i^j)$, 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 node (variable) $X_i$, and $k$ $(k = 1, ..., r_i)$ ranges over all the possible states of $X_i$. Therefore, $x_i^k$ represents the $k$th state of node $X_i$, and $pa_i^j$ is the $j$th parent configuration of node $X_i$.

Given a dataset $D = \{D_1, ..., D_N\}$, which consists of samples of the BN nodes, the goal of parameter learning is to find the most probable values $\hat{\theta}$ for $\theta$ that can best explain the dataset D, which is usually quantified by the log likelihood function $\log(p(D|\theta))$, denoted as $L_D(\theta)$. Assuming that the examples are drawn independently from the underlying distribution, based on the conditional independence assumptions in BNs, we have the log likelihood function in Eq.(1), where $n_{ijk}$ is the count for the case that node $X_i$ has the state $k$, with the state configuration $j$ for its parent nodes.

$$L_D(\theta) = \log \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{n_{ijk}} \tag{1}$$

If the dataset $D$ is complete, ML estimation method can be described as a constrained optimization problem, *i.e*. maximize (Eq.(2)), subject to $n$ equality constrains (Eq.(3)).

$$\text{Max} \qquad L_D(\theta) \tag{2}$$

$$\text{S.T.} \quad g_{ij}(\theta) = \textstyle\sum_{k=1}^{r_i} \theta_{ijk} - 1 = 0 \tag{3}$$

where $g_{ij}$ imposes the constraint that each parameter sums to 1 over all its state, $1 \le i \le n$ and $1 \le j \le q_i$.

If the dataset $D$ is incomplete, ML estimation cannot be applied directly. A common method is standard EM algorithm [6], which starts from some initial point, and then iteratively takes E step and M step to get a local maximum of the likelihood function. Now data is separated into two parts: the observed data $Y$ and missing data $Z$. Assume $\theta^{(0)}$ is the initial point, and $\theta^{(t)}$ denotes the successive estimates after $t$ iteration of E and M steps, $t = 1, 2, ...$, the EM algorithm can be summarized as follows.

*E Step:* compute the expectation of the log likelihood given the observed data Y and the current estimation of parameter $\theta^{(t)}$: $Q(\theta|\theta^{(t)}) = E_{\theta^{(t)}}[\log p(D|\theta)|\theta^{(t)}, Y]$.

*M Step:* find new parameter $\theta^{(t+1)}$ , which maximizes the expected log likelihood computed in the E step: $\theta^{(t+1)} = \arg\max_{\theta} Q(\theta|\theta^{(t)})$.

Particularly for discrete nodes, E step computes the expected counts for all parameters, and M step estimates the parameters by maximizing log likelihood function, given the counts from E step. EM algorithm can guarantee to converge to a local maximum. However, depending on different initializations, it may converge to different local maxima. When there are a large number of missing data, which means there are many local maxima, EM algorithm can get stuck in a local maximum far away from the global one.

### 3.2. Qualitative Constraints

In many real world applications, domain experts usually have an approximated knowledge about the model parameters. This simple and useful qualitative knowledge is often ignored, despite its availability. We introduce two kinds of qualitative constraints, which can be easily specified by domain experts. They are range and relationship constraints.

*Range constraint* defines the upper bound and lower bound of some parameters. Assuming $\alpha_{ijk}$ and $\beta_{ijk}$ are the upper bound and lower bound for parameter $\theta_{ijk}$, then the range constraints can be defined as follows:

$$\beta_{ijk} \leq \theta_{ijk} \leq \alpha_{ijk} \quad \text{where} \quad 0 < \alpha_{ijk} \leq 1 \text{ and } 0 \leq \beta_{ijk} < 1 \tag{4}$$

*Relationship constraint* defines the relative relationship between a pair of parameters. If the two parameters in a relationship constraint share the same node index $i$, and parent configuration $j$, the constraint is called *intra-relationship constraint*, which can be represented as follows:

$$\theta_{ijk} \leq \theta_{ijk'} \text{ where } k \neq k' \tag{5}$$

If the two parameters in a relative relationship constraint do not satisfy the requirement of an intra-relationship constraint, the constraint is called *inter-relationship constraint*. It can be described as follows:

$$\theta_{ijk} \leq \theta_{i'j'k'} \text{ where } i \neq i' \text{ or } j \neq j' \tag{6}$$

### 3.3. Overview of Our Approach

We aim to solve the learning problem by reformulating the problem as a constraint-based optimization problem. Besides the equality constraints implied in BNs, as shown in Eq.(3), we employ two types of qualitative domain knowledge in the form of inequality constraints as discussed in the previous section, denoted as $h_p(x) \leq 0$, where $p$ is the index of the inequality constraints. The parameter learning can therefore be formulated as the optimization problem to find the parameters by:

$$\begin{array}{lll} \text{Max} & L_D(\theta) & \\ \text{S.T.} & g_{ij}(\theta) = \sum_{k=1}^{r_i} \theta_{ijk} - 1 = 0, & 1 \leq i \leq n, \text{ and } 1 \leq j \leq q_i \\ & h_p(\theta) \leq 0, & 1 \leq p \leq S \end{array} \tag{7}$$

where $S$ is the total number of inequality constraints. Using the Lagrange multipliers $\lambda_{ij}$ and $\mu_p$, the objective function to be maximized can be incorporated with the constraints, producing the following augmented objective function

$$f(\theta) = L_D(\theta) - \sum_{i=1}^{n}\sum_{j=1}^{q_i} \lambda_{ij} g_{ij}(\theta) - \sum_{k=1}^{S} \mu_p h_p(\theta) \tag{8}$$

Given Eq.(8), for sparse but complete data, we can directly apply the CML method by maximizing Eq.(8) to estimate the parameters. For incomplete data, we can replace the M step of EM algorithm by the solution to Eq.(8), and iteratively obtain the estimation of the parameters. In the section to follow, we introduce our solution to Eq.(8).

## 4. Parameter Learning With Qualitative Constraints

In this section, we derive the closed form solutions for maximizing Eq.(8) under different types of constraints. While the parameters in $L_D(\theta)$ in Eq.(8) are assumed to be independent on each other, the constraints make them dependent of each other. In practice, we group the parameters into independent sets based on the constraints. Because of the decomposability of the log likelihood function, we can deal with small independent optimization subproblems on independent parameter sets separately instead of dealing with all parameters simultaneously.

For this, we define two kinds of parameter sets: one is the *baseline set*, which contains parameters with the same node and the same parent configuration; the other is the *combined set*, which contains several baseline sets. For example, if two parameters satisfy an intra-relationship constraint as Eq. (5), they are in the same *baseline set*; while two parameters in an inter-relationship constraint as Eq. (6) are in a *combined set*. We first separate parameters into baseline sets, and then if there is a constraint on parameters from different baseline sets, we combine those baseline sets into one new combined set. This process continues until there is no constraint on parameters from different sets. After decomposition of parameters, we solve the constrained optimization subproblems set by set independently.

Specifically, let $Q$ denote a parameter set. Since parameters from one baseline set share the same node $i$ and the same parent configuration $j$, we use $\langle i, j\rangle$ to denote the index of a baseline set. A baseline set can be denoted as $Q = \{\langle i, j\rangle\}$, while a combined set, which consists of several baseline sets, can be denoted as $Q = \{\langle i, j\rangle, \langle i', j'\rangle, ...\}$.

The parameter learning problem can be decomposed into subproblems, one for each set of parameters. A subproblem can be formulated as follows:

$$\begin{array}{ll} \text{Max} & l_D(\theta) = \log \prod_{\langle i,j\rangle \in Q} \prod_{k=1}^{r_i} \theta_{ijk}^{n_{ijk}} \\ \text{S.T.} & g_{ij}(\theta) = \sum_{k=1}^{r_i} \theta_{ijk} - 1 = 0 \text{ for } \langle i, j\rangle \in Q \\ & h_p(\theta) \leq 0 \ \ for\ 1 \leq p \leq S_Q \end{array} \tag{9}$$

where $g_{ij}$ represents an equality constraint, $h_p$ represents an inequality constraint, $S_Q$ is the number of inequality constraints in set $Q$.

Since the log likelihood function is concave, and the qualitative constraints are linear, Karush-Kuhn-Tucker (KKT) conditions [16] become sufficient to determine the solution to Eq.(9). The KKT conditions for the problem described in Eq.(9) are:

$$\begin{array}{l} \nabla_\theta [l_D(\theta) - \sum_{\langle i,j\rangle \in Q} \lambda_{ij} g_{ij}(\theta) - \sum_{p=1}^{S_Q} \mu_p h_p(\theta)] = 0, \\ \\ \begin{array}{rll} g_{ij}(\theta) = 0, & \text{for} & \langle i, j\rangle \in Q \\ h_p(\theta) \leq 0, & \text{for} & 1 \leq p \leq S_Q \\ \mu_p \geq 0, & \text{for} & 1 \leq p \leq S_Q \\ \mu_p * h_p(\theta) = 0, & \text{for} & 1 \leq p \leq S_Q \end{array} \end{array} \tag{10}$$

In optimization, an inequality constraint $h_p \leq 0$ is active if $h_p = 0$, or inactive if $h_p < 0$. Based on this theory, we will derive closed form solutions for each type of constraints.

### 4.1. Range Constraints

Since range constraints (Eq.(4)) are applied to every individual parameters, we can solve the subproblems with range constraints within baseline sets. There are two constraints for each parameter $\theta_{ijk}$ in a baseline set $Q = \{\langle i,j\rangle\}$: $h_k^\alpha(\theta) = \theta_{ijk} - \alpha_{ijk} \leq 0$ (upper bound constraint), and $h_k^\beta(\theta) = \beta_{ijk} - \theta_{ijk} \leq 0$ (lower bound constraint).

As the objective function is concave and the range constraints are linear, the maximum solution either lies inside the feasible region defined by all constraints, when no constraint is active, or on the boundary defined by the active constraints, when some of the constraints are active. Assuming $K_Q^\beta$ and $K_Q^\alpha$ are the sets of active constraints for lower bound and upper bound of parameters in $Q$ respectively, and $K_Q = K_Q^\beta \cup K_Q^\alpha$ represents the set for all active constraints of parameters in $Q$, then the closed form solution for $\theta_{ijk}$ is as follows:

$$\theta_{ijk} = \begin{cases} \beta_{ijk} & \text{if } k \in K_Q^\beta \\ \alpha_{ijk} & \text{if } k \in K_Q^\alpha \\ (1 - \sum_{k \in K_Q^\beta} \beta_{ijk} - \sum_{k \in K_Q^\alpha} \alpha_{ijk}) \frac{n_{ijk}}{\sum_{k \notin K_Q} n_{ijk}} & \text{otherwise} \end{cases} \tag{11}$$

The derivation is as follows. From the first equation of KKT conditions (Eq.(10)), we obtain $\theta_{ijk} = \frac{n_{ijk}}{\lambda_{ij} - \mu_k^\alpha + \mu_k^\beta}$. Because $\theta_{ijk}$ cannot be greater than $\alpha_{ijk}$ and less than $\beta_{ijk}$ at the same time, at most one of the upper bound constraint $h_k^\alpha$ and lower bound constraint $h_k^\beta$ for a parameter $\theta_{ijk}$ can be active at a time. Based on whether there is an active constraint for $\theta_{ijk}$, two cases are considered.

- Case 1: If one of the upper bound and lower bound constraints is active, then $\theta_{ijk} = \alpha_{ijk}$, when $h_k^\alpha(\theta) = 0$; and $\theta_{ijk} = \beta_{ijk}$, when $h_k^\beta(\theta) = 0$.
- Case 2: If range constraints are not active, then $h_k^\alpha(\theta) < 0$, $h_k^\beta(\theta) < 0$ and $\mu_k^\alpha = \mu_k^\beta = 0$. Hence $\theta_{ijk} = \frac{n_{ijk}}{\lambda_{ij}}$. Summing up over all parameters whose constraints are not active, we get: $(1 - \sum_{k \in K_Q^\beta} \beta_{ijk} - \sum_{k \in K_Q^\alpha} \alpha_{ijk}) = \sum_{k \notin K_Q} \theta_{ijk} = \frac{\sum_{k \notin K_Q} n_{ijk}}{\lambda_{ij}}$. Thus, we can obtain $\lambda_{ij} = \frac{\sum_{k \notin K_Q} n_{ijk}}{1 - \sum_{k \in K_Q^\beta} \beta_{ijk} - \sum_{k \in K_Q^\alpha} \alpha_{ijk}}$, and $\theta_{ijk} = (1 - \sum_{k \in K_Q^\beta} \beta_{ijk} - \sum_{k \in K_Q^\alpha} \alpha_{ijk}) \frac{n_{ijk}}{\sum_{k \notin K_Q} n_{ijk}}$, as shown in Eq.(11).

In this way, we derive the closed form solution for range constraints. To obtain solution in Eq.(11), we need to identify active constraints. Table 1 summarizes the algorithm to find active range constraints. The main idea of this algorithm is to search for the active constraints using the criteria in Eq.(12). Due to the page limit, we do not provide the proof for this equation.

Table 1. Algorithm for finding active range constraints

| | |
|---|---|
| Step 1: | Check the consistency of the range constraints: $0 < \alpha_{ijk} \leq 1$, $0 \leq \beta_{ijk} < 1$, $\alpha_{ijk} > \beta_{ijk}$, $\sum_{k=1}^{r_i} \beta_{ijk} \leq 1$, and $\sum_{k=1}^{r_i} \alpha_{ijk} \geq 1$ for $1 \leq k \leq r_i$. If satisfied, continue; else change constraints. |
| Step 2: | If $\sum_{k=1}^{r_i} \alpha_{ijk} = 1$, all the upper bound constraints should be active; else if $\sum_{k=1}^{r_i} \beta_{ijk} = 1$, all the lower bound constraints should be active; else, continue. |
| Step 3: | Perform the ML estimation of parameters without constraints. Check the constraints with the estimated parameters $\theta_{ijk}^* = \frac{n_{ijk}}{N_{ij}}$. If no constraint is violated, then there is no active range constraint; else, continue. |
| Step 4: | List all possible combinations of active constraints, and rule out the combination if it contains more than $r_i - 1$ active constraints or $\sum_{k \in K_Q^\beta} \beta_{ijk} + \sum_{k \in K_Q^\alpha} \alpha_{ijk} \geq 1$. |
| Step 5: | For each of the remaining combinations, compute $\lambda_{ij}$, until finding a $\lambda_{ij}$ satisfying the criteria in Eq.(12). |

$$\begin{cases} \lambda_{ij} \leq \frac{n_{ijk}}{\alpha_{ijk}} & k \in K_Q^\alpha \\ \lambda_{ij} \geq \frac{n_{ijk}}{\beta_{ijk}} & k \in K_Q^\beta \\ \lambda_{ij} \geq \frac{n_{ijk}}{\alpha_{ijk}},\ \lambda_{ij} \leq \frac{n_{ijk}}{\beta_{ijk}} & otherwise \end{cases} \tag{12}$$

### 4.2. Intra-Relationship Constraints

An Intra-relationship constraint defines the relationship between two parameters within one baseline set. Assuming parameters within one baseline set $Q = \{\langle i,j\rangle\}$ are $\theta_{ij1}, ...\theta_{ijr_i}$, which can be partitioned into $Q = A \cup B \cup C$, where $A = \{a_p | p = 1, 2, ..., S_Q\}$, $B = \{b_p | p = 1, 2, ..., S_Q\}$, such that $h_p(\theta) = \theta_{ija_p} - \theta_{ijb_p} \leq 0$, for $1 \leq p \leq S_Q$, and $C$ is the set of parameters without intra-relationship constraints, the closed form solution for parameter $\theta_{ijk}$is as follows:

$$\theta_{ijk} = \begin{cases} \frac{n_{ija_p} + n_{ijb_p}}{2N_{ij}} & if\ k = a_p \text{ or } b_p \text{ and } n_{ija_p} \geq n_{ijb_p} \\ \frac{n_{ijk}}{N_{ij}} & Otherwise \end{cases} \tag{13}$$

where $N_{ij} = \sum_{k=1}^{r_i} n_{ijk}$. The derivation is similar to Niculescu *et al*. [18].

### 4.3. Inter-Relationship Constraints

An Inter-relationship constraint defines the constraint applied on two parameters $\theta_{i'j'a}$ and $\theta_{i''j''b}$ from different baseline sets $Q_A$ and $Q_B$, thus the subproblem for parameters with an inter-relationship constraint is applied on a combined parameter set $Q = Q_A \cup Q_B$, where baseline set $Q_A = \{\langle i', j'\rangle\}$ and baseline set $Q_B = \{\langle i'', j''\rangle\}$, such that $h(\theta) = \theta_{i'j'a} - \theta_{i''j''b} \leq 0$. Let $N_A = \sum_{\langle i,j\rangle \in Q_A} n_{ijk}$, $N_B = \sum_{\langle i,j\rangle \in Q_B} n_{ijk}$, $n_a = n_{i'j'a}$, and $n_b = n_{i'j'b}$. The closed form solution for parameters with inter-relationship constraint is as follows. If $n_a N_B - N_A n_b \geq 0$

$$\theta_{ijk} = \begin{cases} \frac{n_a + n_b}{N_A + N_B} & ijk = i'j'a \text{ or } i''j''b \\ (1 - \frac{n_a + n_b}{N_A + N_B}) \frac{n_{ijk}}{N_A - n_a} & \langle i,j\rangle \in Q_A \text{ and } k \neq a \\ (1 - \frac{n_a + n_b}{N_A + N_B}) \frac{n_{ijk}}{N_B - n_b} & \langle i,j\rangle \in Q_B \text{ and } k \neq b \end{cases} \tag{14}$$

Else $$\theta_{ijk} = \begin{cases} \frac{n_{ijk}}{N_A} & \langle i,j\rangle \in Q_A \\ \frac{n_{ijk}}{N_B} & \langle i,j\rangle \in Q_B \end{cases}$$

The brief derivation of the solution is as follows. The KKT conditions are:

$$\nabla_\theta[l_D(\theta) - \lambda_A g_A(\theta) - \lambda_B g_B(\theta) - \mu h(\theta)] = 0$$

$$\begin{array}{ccc} g_A(\theta) = 0 & g_B(\theta) = 0 & \\ h(\theta) \leq 0 & \mu \geq 0 & \mu * h(\theta) = 0 \end{array} \quad (15)$$

From the first equation of KKT conditions (Eq.(15)), we can obtain:

$$\theta_{ijk} = \begin{cases} \frac{n_{ijk}}{\lambda_A + \mu} & ijk = i'j'a \\ \frac{n_{ijk}}{\lambda_B - \mu} & ijk = i''j''b \\ \frac{n_{ijk}}{\lambda_A} & \langle i,j \rangle \in Q_A \text{ and } k \neq a \\ \frac{n_{ijk}}{\lambda_B} & \langle i,j \rangle \in Q_B \text{ and } k \neq b \end{cases} \quad (16)$$

Two cases are considered, depending on whether the inter-relationship constraint is active or not:

- Case 1:$h(\theta) = 0$ and $\mu \geq 0$
  We can solve $\lambda_A$, $\lambda_B$, and $\mu$ with the following equations:
  $$\begin{cases} \frac{n_a}{\lambda_A + \mu} = \frac{n_b}{\lambda_B - \mu} = \frac{n_a + n_b}{\lambda_A + \lambda_B} \\ \frac{n_a}{\lambda_A + \mu} + \frac{N_A - n_a}{\lambda_A} = 1 \\ \frac{n_b}{\lambda_B - \mu} + \frac{N_B - n_b}{\lambda_B} = 1 \end{cases} \quad (17)$$
  The first equation is $h(\theta) = 0$, the second and the third are from $g_A(\theta) = 0$, and $g_B(\theta) = 0$. Also, from $\mu \geq 0$, we can get $n_a N_B - N_A n_b \geq 0$. In this way, we obtain the first part of closed form solution (Eq.(15)).
- Case 2: $h(\theta) < 0$ and $\mu = 0$
  It is equivalent to the case that no inequality constraints are applied. From $g_A(\theta) = 0$, we can get $\lambda_A = \sum_{\langle i,j \rangle \in Q_A} n_{ijk} = N_A$. Similarly, we can get $\lambda_B = N_B$. Plug them into Eq.(15), we can obtain the second part of closed form solution(Eq.(15)). From $h(\theta) < 0$, we get $n_a N_B - N_A n_b < 0$.

# 5. Evaluation with Synthetic Data

## 5.1. Experimental Design

In order to test the performance of our method against ML estimation and the standard EM algorithm given sparse data and incomplete data respectively, we design the experiment as follows.

*A. Models:* We test the algorithms on multiple BNs with the same number of nodes of 20, but with different randomly generated initial parameters and structures. For one specific BN structure, 11 BNs with different initializations of parameters are generated. One of them is treated as the ground truth, and 10 others as different initializations for parameter learning. A sample BN is shown in Figure 1.

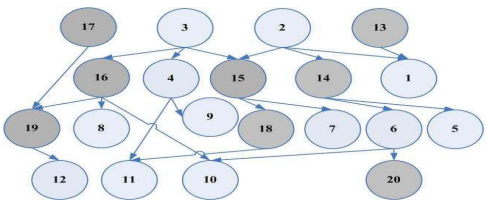



Figure 1. One BN example: the shaded nodes (node 13 to 20) are hidden nodes, the others are observable nodes.

*B. Training and Testing Data:* For the case of sparse data, 700 samples are generated from the ground truth BN, 200 for testing and the remaining 500 for training. The training data are divided into 10 datasets for 10 different initializations, each with 50 samples. For the case of incomplete data, 400 samples are drawn from the ground truth BN, half for training, half for testing, and all training data associated with hidden nodes are removed.

*C. Constraints:* For the case of sparse data, we randomly choose a subset of parameters from all parameters, and impose constraints on the selected parameters. For the case of incomplete data, we randomly choose parameters from only those of the hidden nodes, and impose constraints on them. The number of constraints in a CPT is no more than 2. To guarantee the correctness of the constraints, all the constraints are defined based on the ground truth.

*D. Performance Evaluation:* Two criteria are used for evaluation: Kullback-Leibler (K-L) divergence, which measures the distance between the learned parameters and the ground truth, and negative log likelihood, which measures how well a learned BN matches with the testing data.

## 5.2. Sparse Data

With complete but sparse data, we compare the learning performance of ML estimation with our method with range constraints, intra-relationship constraints and inter-relationship constraints respectively, as shown in Figure 2. The first column of Figure 2 compares ML estimation with CML w.r.t. K-L divergence. The x-coordinate and the y-coordinate represent node index and K-L divergence respectively. Since the learning results are from 10 datasets, we plot the mean and standard deviation by the median of each bar and the height of the bar respectively. We can see that CML is better than ML estimation in both mean and standard deviation of KL-divergence. More specifically, the mean K-L divergence for ML estimation is 0.2087, which decreases to 0.0786 for CML with range constraints, 0.1763 for CML with intra-relationship constraints, and 0.1546 for CML with inter-relationship constraints.

The second column of Figure 2 shows the comparison results w.r.t. negative log likelihood. The x-coordinate denotes the index of dataset, and the y-coordinate denotes the negative log likelihood value. We can see that for each dataset, CML has better negative log likelihood value than ML estimation. The mean negative log likelihood value is 2526 for ML, which decreases to 2394 for CML with range constraints, 2490 for CML with intra-relationship constraints, and 2492 for CML with inter-relationship constraints.

## 5.3. Incomplete Data

With incomplete data, we compare the learning performance of our method with standard EM method as shown in Figure 3. Since 10 initializations of parameters are used, we plot the mean and standard deviation. As only the parameters for hidden nodes (node 13 to 20) may have constraints,

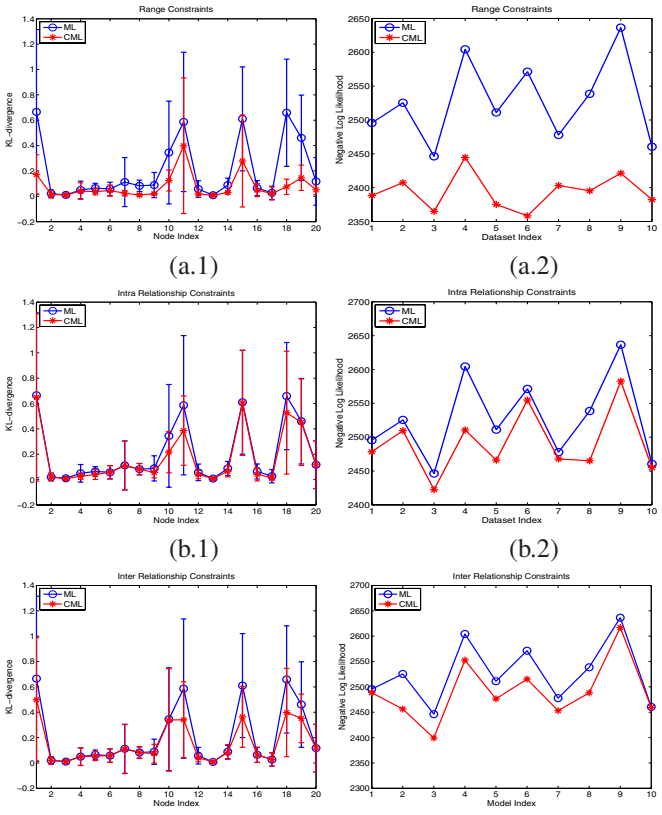



Figure 2. Sparse data learning results comparisons: ML estimation vs. CML. (a) range constraints; (b) intra-relationship constraints; (c) inter-relationship constraints. The left column are the comparisons w.r.t. K-L divergence, and the right with negative log likelihood.

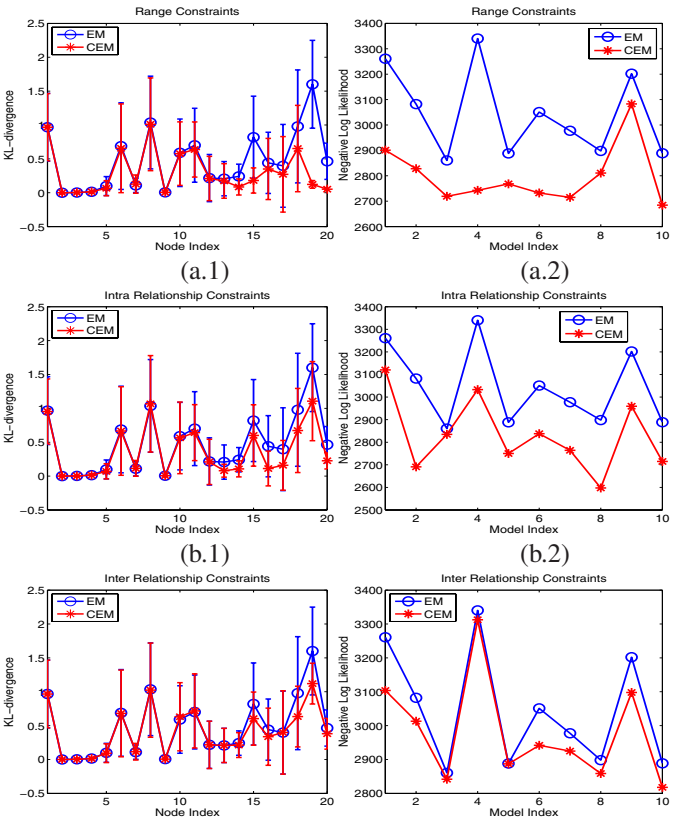



Figure 3. Incomplete data learning results comparisons: EM vs. CEM. (a) range constraints; (b) intra-relationship constraints; (c) inter-relationship constraints. The left column are the comparisons w.r.t. K-L divergence, and the right with negative log likelihood. Node 13 to 20 are hidden nodes, which may have constraints.

we can see clearly from the figures in the left column that those nodes benefit most from the constraints. The average K-L divergence of hidden nodes decreases from 0.6437 for EM to 0.2361 for CEM with range constraints, 0.3830 for CEM with intra-relationship constraints, and 0.4864 for CEM with inter-relationship constraints. The right column of figures show that CEM is better than EM for each initialization of BN model. The average negative log likelihood decreases from 3045 for EM to 2798 for CEM with range constraints, 2978 for CEM with intra-relationship constraints and 2830 for CEM with inter-relationship constraints.

## 6. Facial Action Unit Recognition

In this section, we apply our method to facial action unit (AU) recognition [20]. The Facial Action Coding System (FACS) [8] is the most commonly used system for facial behavior analysis. Based on FACS, facial behaviors can be decomposed to a set of AUs, each of which is related to the contraction of a specific set of facial muscles. An automatic AU recognition system has many applications. Developing such a system requires manually labeling AUs in each training image, which is usually done by certified AU coders. However, constructing a large amount of reliably labeled AU images, especially for spontaneous facial expressions, is very difficult since training AU coder and manually scoring the AUs are expensive and time consuming. The proposed method can significantly reduce our dependence on the manually labeled images.

As shown in [20], there are semantic relationships among AUs. Instead of recognizing each AU individually, a BN is employed to explicitly model the probabilistic relationships among AUs. Specifically, there are two types of relationships among AUs. Type I is *cooccurrence relationship*, which means some AUs appear simultaneously. For example, AU6 (cheek raiser) tends to happen with AU12 (lip corner puller), when smiling. Type II is *mutual exclusive relationship*, which means it is nearly impossible for some AUs to appear together. For instance, AU25 (lips part) can hardly happen with AU24 (lip presser) simultaneously.

Following the work in [20], we use a BN as shown in Figure 4 to capture the semantic relationships among the 14 frequently occurring AUs. The larger circular nodes in the model represent AUs while the smaller nodes represent their image measurements. The measurement are acquired

using an image-based technique [2]. Tong et al [20] have demonstrated that the BN model is superior to the state of the art AU recognition methods. But they use a very large set of data to train their BN model. We will show that we can achieve the comparable results using only a fraction of their training data.

Every link between two AU nodes has a sign provided by the domain expert. The "+" sign denotes positive influence, which means two AU nodes have cooccurrence relationship, and one of them plays the dominant role. For example, it is difficult to do AU2 (outer brow raiser) without performing AU1 (inner brow raiser), but we can do AU1 without AU2, so AU2 has a positive influence on AU1. On the other hand, "-" denotes negative influence, which means the two AU nodes have mutual exclusive relationship.

As the manually labeled data is often limited, we want to use sparse data to learn the parameters, with the help of qualitative constraints. Although it is hard to obtain manually labeled data for AU nodes, it is easier to obtain the measurements of AUs by some computer vision techniques. Therefore, we want to learn the parameters also using the measurements of AUs. For this purpose, 14 image measurement nodes are introduced to represent the AU measurement results. Now, there are two kinds of links in the BN: the links between AU nodes denote the relationship between AUs; the links between AU nodes and the corresponding measurement nodes represent the uncertainties in AU measurement.

We extract constraints based on the following rules provided by domain experts:

*1. Intra-Relationship Constraint*: In spontaneous cases, some AUs rarely occur. One example for this case is AU27, and the rule is $P(AU27 = 1) \leq P(AU27 = 0)$, where 1 means presence and 0 means absence.

*2. Inter-Relationship Constraint I*: Considering an AU node $AU_i$ has only one parent node $AU_j$, if the sign of the link is positive, we have $P(AU_i = 1|AU_j = 0) \leq P(AU_i = 1|AU_j = 1)$, *e.g.* $P(AU1 = 1|AU2 = 0) \leq P(AU1 = 1|AU2 = 1)$; if the sign of the link is negative, then we can get $P(AU_i = 1|AU_j = 1) \leq P(AU_i = 1|AU_j = 0)$, *e.g.* $P(AU6 = 1|AU27 = 1) \leq P(AU6 = 1|AU27 = 0)$.

*3. Inter-Relationship Constraint II*: Considering an AU node $AU_i$ has more than one AU parent nodes, $AU^P$ denote all the parent nodes with positive links, and $AU^N$ denote all the parent nodes with negative links. Then we get $P(AU_i = 1|AU^P = 0, AU^N = 1) \leq P(AU_i = 1|AU^P = 1, AU^N = 0)$, *e.g.* $P(AU15 = 1|AU24 = 0, AU25 = 1) \leq P(AU15 = 1|AU24 = 1, AU25 = 0)$.

*4. Range Constraint I*: If an AU node $AU_i$ has more than one parent nodes $AU^P$, and all of them with positive influence, then $P(AU_i = 1|AU^P = 1) \geq 0.8$.

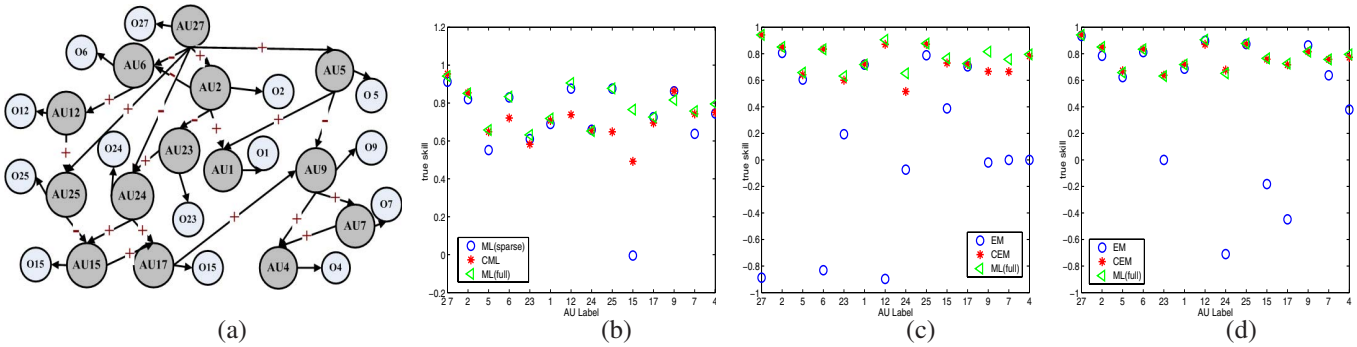
*5. Range Constraint II*: If an AU node $AU_i$ has more than one parent nodes $AU^N$, and all of them with negative influence, then $P(AU_i = 1|AU^N = 1) \leq 0.2$.

*6. Range Constraint III*: For each measurement node, a domain expert can provide the range of the performance of the classifier, such as $P(O_i = 1|AU_i = 1)$ and $P(O_i = 0|AU_i = 0)$, which represent the accuracy of the corresponding classifier.

Please note the above constraints are due to either facial anatomy or due to certain facial patterns. They are generic enough to be applied to different databases and to different individuals.

The 8000 images used in experiments are collected from Cohn and Kanade's DFAT-504 database [15], 80% data for training and 20% for testing. Training data are used for learning the parameters in the BN (Figure 4 (a)), while testing data are used to perform AU recognition through inference given learned BN. We consider 3 cases of training data: 1. 300 complete training data for both AU and measurement nodes; 2. Full training data ($80\% \times 8000$) for measurement nodes, but no data for AU nodes; 3. 300 data for AU nodes, and full data for measurement nodes. In case 1, all nodes are observable, *i.e.* data are complete, thus CML and ML estimation methods are compared. In cases 2 and 3, nodes are partially observable, *i.e.* data are incomplete, therefore, EM and CEM are compared.

Figure 4 (b), (c), and (d) show the recognition results for the three cases respectively. The x and y coordinates represent the AU node index and the true skill score (the difference between true positive rate and false positive) respectively. We can see from Figure 4 (b) that ML estimation with 300 data fails at AU15, but CEM makes significant improvement at AU15. The average true skill score for all AU nodes increases from 0.6993 for ML estimation to 0.7182 for CML with sparse data. Figure 4 (c) shows when there are no data for AU nodes, though with full training data for measurement nodes, EM fails with the average true skill score of 0.1067. In contrast, CEM is significantly better than EM with average true skill score of 0.7439. From Figure 4 (d), we can see that the average true skill score for EM is 0.4394, which is better than 0.1067 (no data for AU nodes). However it is much less than 0.7182 (ML estimation with 300 data), which means although EM has additional data from measurement nodes than ML estimation, it cannot make full use of them. On the other hand, the average true skill score for CEM here is 0.7786, which is the best of three cases (b,c,d). It means that with qualitative constraints, CEM can make full use of data to improve the learning accuracy. Additionally, for comparison, we also evaluate ML estimation with full training data. Its average true skill score is 0.7808, which is slightly better than CEM in case 3. These results are extremely encouraging, as using our methods with domain-specific yet generic qualitative constraints, and with a small number of manually labeled data (300), we can achieve similar learning accuracy




(a) (b) (c) (d)

Figure 4. Comparison of average AU recognition results using the BNs learned from MLE, EM, and CEM respectively (a) BN for AU recognition; (b) 300 complete data; (c) full training data for measurement nodes, but no data for AU nodes; (d) 300 data for AU nodes, full training data for measurement nodes.

to the ML estimation with full training data (6400).

## 7. Conclusion

Qualitative domain knowledge generally exists in computer vision. We define two types of constraints to represent the qualitative domain knowledge, and derive closed form solution for the ML BN parameter estimation with the two types of constraints respectively. For the case of sparse data, we directly apply our constrained maximum likelihood estimator, while for incomplete data, we extend EM method by replacing M step with our constrained maximum likelihood estimator. We further apply our method to a BN for AU recognition. The experimental results from both synthetic data and real data demonstrate that our method can fully exploit the domain knowledge to improve parameter learning accuracy. The proposed methods can significantly reduce our dependance on the labeled data and they can be applied to many computer vision tasks including segmentation, tracking, image retrieval, and object recognition.

## References


[1] E. E. Altendorf, A. C. Restificar, and T. G. Dietterich. Learning from sparse data by exploiting monotonicity constraints. In *UAI*, pages 18–26, 2005. 2

[2] M. S. Bartlett, G. Littlewort, M. G. Frank, C. Lainscsek, I. Fasel, and R. Movellan. Recognizing facial expression: Machine learning and application to spontaneous behavior. *CVPR*, 2:568–573, 2005. 7

[3] J. Binder, D. Koller, S. Russell, and K. Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, pages 213–244, 1997. 2

[4] I. Cohen, N. Sebe, L. Chen, A. Garg, and T. S. Huang. Semisupervised learning of classifiers: Theory, algorithms, and their application to human-computer interaction. *Computer Vision and Image Understanding*, 91:160–187, 2003. 1

[5] C. P. de Campos and F. G. Cozman. Belief updating and learning in semi-qualitative probabilistic networks. In *UAI*, 2005. 2

[6] 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

[7] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods In Practice*. Springer-Verlag, New York, 2001. 2

[8] P. Ekman and W. Friesen. *Facial Action Coding System: A Technique for the Measurement of Facial Movement*. Consulting Psychologists Press, 1978. 6

[9] G. Elidan and N. Friedman. The information bottleneck EM algorithm. *UAI*, pages 200–209, 2003. 2

[10] G. Elidan, M. Ninio, N. Friedman, and D. Schuurmans. Data perturbation for escaping local maxima in learning. *AAAI*, pages 132–139, 2002. 2

[11] A. Feelders and L. van der Gaag. Learning Bayesian network parameters under order constraints. pages 37–53, 2006. 2

[12] D. Geiger and D. Heckerman. A characterization of the Dirichlet distribution through global and local parameter independence. *The Annals of Statistics*, 25:1344–1369, 1999. 1

[13] D. Heckerman. A tutorial on learning with Bayesian networks. In M. Jordan, editor, *Learning in Graphic Models*. MIT Press, Cambridge, MA, 1999. 1

[14] D. Hoiem, A. A. Efros, and M. Hebert. Putting objects in perspective. *CVPR*, 2006. 1

[15] T. Kanade, J. F. Cohn, and Y. Tian. Comprehensive database for facial expression analysis. *Proc. of FG00*, 2000. 7

[16] H. W. Kuhn and A. W. Tucker. Nonlinear programming. *Proc. of the second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492. 3

[17] E. N. Mortensen and J. Jia. Real-time semi-automatic segmentation using a Bayesian network. *CVPR*, 2006. 1

[18] R. S. Niculescu, T. M. Mitchell, and R. B. Rao. A theoretical framework for learning Bayesian networks with parameter inequality constraints. In *IJCAI*, 2007. 2, 4

[19] P. Nillius, J. Sullivan, and S. Carlsson. Multi-target tracking - linking identities using Bayesian network inference. *CVPR*, 2006. 1

[20] Y. Tong, W. Liao, and Q. Ji. Inferring facial action units with causal relations. *CVPR*, 2006. 1, 6, 7

[21] F. Wittig and A. Jameson. Exploiting qualitative knowledge in the learning of conditional probabilities of Bayesian networks. *UAI*, pages 644–652, 2000. 2