Improving Bayesian Network Parameter Learning using Constraints

Cassio P. de Campos
Rensselaer Polytechnic Institute
decamc@rpi.edu

Qiang Ji
Rensselaer Polytechnic Institute
jiq@rpi.edu

Abstract

This paper describes a new approach to unify constraints on parameters with training data to perform parameter estimation in Bayesian networks of known structure. The method is general in the sense that any convex constraint is allowed, which includes many proposals in the literature. Driven by a maximum entropy criterion and the Imprecise Dirichlet Model, we present a constrained convex optimization formulation to combine priors, constraints and data. Experiments indicate benefits of this framework.

1 Introduction

Bayesian Networks (BNs) encode a joint probability distribution for a set of random variables in a compact graph structure. The problem of parameter learning concerns the estimation of probability measures of conditional probability distributions, given the graph structure of the BN. Many techniques depend heavily on training data. Ideally, with enough data, it is possible to learn parameters by standard statistical analysis like maximum likelihood (ML) estimation. However, data may be insufficient, leading to inaccurate estimations.

This paper proposes a framework for the parameter learning problem that combines data and domain knowledge in the form of constraints. There are two types of constraints: *soft constraints* on priors and *hard constraints* on estimations. Driven by the Imprecise Dirichlet Model [17], prior beliefs are encoded using a set of Dirichlet distributions. Combined with data and constraints on estimations, the result is a set of estimations, on which we apply the maximum entropy principle to obtain a single estimation. Constraints on estimations are viewed as *hard* constraints and assumed to be correct. Thus only general and certainly valid constraints shall be used. On the other hand, constraints on priors are *soft* because estimations are adapted and corrected by training data even if some *soft* constraints are wrongly stated. Altogether, we can encode constraints that are certain as well as less reliable constraints.

There are many approaches to parameter learning of BNs using constraints. For instance, penalty functions can be employed [1], but global optimality is not always guaranteed. Isotonic regression is also applicable, but its complexity is high [7]. Non-convex optimization also leads to high complexities [5]. Closed-form solutions were investigated [9, 10], but they do not allow overlap of constraints (the same parameters may not appear in different constraints). Because of that, many types of constraints can not be represented. The framework presented here tries to overcome such limitations.

2 Problem definition

A Bayesian network is a triple $(G, \mathcal{X}, \mathcal{P})$, where G is a directed acyclic graph with n nodes associated to discrete random variables $\mathcal{X}$ (a variable per node), and $\mathcal{P}$ is a collection of parameters $\theta_{ijk} = p(x_i^k | pa_i^j)$, with $\sum_k \theta_{ijk} = 1$, where x_i^k is a value or state of X_i and pa_i^j a complete instantiation for the parents PA_i of X_i in G (it represents a set of states for PA_i). In a BN every variable is conditionally independent of its non-descendants given its parents (Markov condition). Thus the joint probability distribution is obtained by $p(X_1, \ldots, X_n) = \prod_i p(X_i | PA_i)$. We focus on parameter learning in a BN where its structure is known in advance. Given a data set D where each element is a sample of the BN variables, the goal of parameter learning is to find the most probable values for the vector θ, which can be quantified by the log likelihood function $L_D(\theta) = \log(p(D|\theta))$. Assuming that samples are drawn independently from the underlying distribution, we need to maximize $L_D(\theta) = \log \prod_{ijk} \theta_{ijk}^{n_{ijk}}$, where n_{ijk} indicates how many elements of D contain both x_i^k and pa_i^j. Maximum likelihood estimation has its optimum at $\theta_{ijk} = \frac{n_{ijk}}{\sum_k n_{ijk}}$.

Another usual parameter learning technique is the Dirichlet model, where one starts by assuming that an

expert has specified a prior BN, denoted by $\mathcal{N}_p$, that conveys her prior beliefs. The goal is to learn the parameters of multinomial distributions on θ_{ij} using both $\mathcal{N}_p$ and data. The Dirichlet distribution is a natural parametric model for $p(\theta_{ij})$, because it is conjugate with the multinomial distribution. A possible parametrization is $p(\theta_{ij}) \propto \prod_k \theta_{ijk}^{s\tau_{ijk}-1}$ for $s \geq 0$ and $\sum_k \tau_{ijk} = 1$, where the hyper-parameter s controls dispersion and hyper-parameters τ_{ijk} control location [17]. The parameter s is often interpreted as the *size* of a database encoding the same prior beliefs as the Dirichlet distribution. We assume that $\mathcal{N}_p$ is associated with a single positive number s that encodes the *quality* of the prior BN, and that parameters of $\mathcal{N}_p$ define τ such that τ_{ijk} corresponds to θ_{ijk} in the prior. Then, using expectation as estimator, the optimal estimate θ_{ijk} is the posterior expected value: $\theta_{ijk} = \frac{s\tau_{ijk}+n_{ijk}}{s+\sum_k n_{ijk}}$.

Standard estimation methods are usually enough when there are enough data. However, when a small amount of data is available, they may produce unreliable estimations. A way to improve estimations is through the use of constraints. Let θ_A be a sequence of parameters, α_A a corresponding sequence of constant numbers and α also a constant. A *linear relationship constraint* is defined as

$$\sum_{\theta_{ijk} \in \theta_A, \alpha_{ijk} \in \alpha_A} \alpha_{ijk} \cdot \theta_{ijk} \leq \alpha, \qquad (1)$$

that is, any linear constraint over parameters can be expressed as a *linear relationship constraint*. For instance, qualitative influences and synergies [18] can be expressed by linear constraints. Suppose X_1, X_2, X_3 are random variables that assume values in $\{x_i^1, x_i^2\}$, with x_i^2 greater than x_i^1, such that X_1 is the child of X_2 and X_3. It is said that X_2 has a positive influence on X_1 if $p(x_1^2|x_2^2, x_3^1) \geq p(x_1^2|x_2^1, x_3^1)$[1] and $p(x_1^2|x_2^2, x_3^2) \geq p(x_1^2|x_2^1, x_3^2)$, that is, a greater value of X_2 increases the probability of a greater value of X_1 (for both values of X_3). A positive synergy of two parents in a common child happens when the parents influence the child together, for example, $p(x_1^2|x_2^1, x_3^1) + p(x_1^2|x_2^2, x_3^2) \geq p(x_1^2|x_2^1, x_3^2) + p(x_1^2|x_2^2, x_3^1)$. This means that a greater value of the child is more likely when the parents have the same value. In fact these are just simple (but important) examples of constraints that are allowed. Other examples are sum of parameters, range, relationship, and ratio constraints [9], weak and strong, monotonic and non-monotonic influences and synergies [11], among many others. Our assumption about constraints is even more general: they must define a (possibly non-linear) convex parameter space, that is, any constraint in the form $h(\theta) \leq 0$, where h is convex, is allowed. Most of recent literature in this topic can be expressed by convex constraints [9, 10, 11, 18]. Such flexibility allows for a better description of the knowledge, as we have no restriction regarding the number of times a parameter appear in constraints or whether constraints involve distinct distributions of the BN.

[1] We use a probability notation for ease of expose, as each parameter θ_{ijk} is a probability value of the network.

3 Parameter learning using convex optimization

If we only have constraints on θ, ML can be solved by convex programming, as a maximization of a concave log-likelihood function must be solved. There are optimization algorithms to solve convex programming in polynomial time [2, 4]. They can be as fast as linear programming solvers. Convex programming has the attractive property that any local optimum is also a global optimum [4]. This idea of constrained ML is rather intuitive in its interpretation and in the method to solve it. However ML does not allow us to define prior assessments (probabilities cannot be directly taken as constraints because if they are fixed at this stage, the empirical data cannot change them). The interpretation of constraints only as *hard* constraints on estimations is eventually too inflexible, and in some cases it is more profitable to interpret expert's belief as a *partial assessment of a prior distribution*.

Besides constraints on parameters θ_{ijk}, we allow constraints on hyper-parameters τ_{ijk}. The idea is to work with both constraints on θ and constraints on τ. So, assume that an expert has specified two sets of constraints denoted by $\mathcal{C}_p$ and $\mathcal{C}$, that conveys her prior beliefs and some knowledge about parameters, respectively. The content of $\mathcal{C}_p$ is viewed as the set of *constraints on the hyper-parameters* τ_{ijk} of Dirichlet distributions for a fixed value of s, while $\mathcal{C}$ is the set of *hard* constraints on estimations. Constraints on $\mathcal{C}_p$ are *constraints on the prior*, not in the probability values themselves. The only assumption for constraints on $\mathcal{C}_p$ is that they must be convex constraints on τ (this is same assumption as for $\mathcal{C}$). If the expert is certain, she defines a constraint over θ. Otherwise, a similar constraint, but now over τ, may be used. In summary, constraints of $\mathcal{C}_p$ and $\mathcal{C}$ can be specified similarly. The former defines restrictions on parameters τ, while the latter defines restrictions for θ. This formulation is based on the *Imprecise Dirichlet Model* [17], which has received great attention recently [5, 14].

The result is a set of distributions that must satisfy

$\mathcal{C}(\theta), \mathcal{C}_p(\tau)$ and equations

$$\theta_{ijk} = \frac{s\tau_{ijk} + n_{ijk}}{s + \sum_k n_{ijk}}, \quad (2)$$

where n_{ijk} are the counts from the data set (simplex constraints $\forall_{ij} \sum_k \theta_{ijk} = 1$ and $\forall_{ij} \sum_k \tau_{ijk} = 1$ are assumed to be in $\mathcal{C}(\theta)$ and $\mathcal{C}_p(\tau)$). Equation (2) is the *glue* between $\mathcal{C}(\theta)$ and $\mathcal{C}_p(\tau)$. This formulation has several attractive features. First, it deals with qualitative and numerical aspects in a uniform manner. Second, it uses constraints on priors and on estimations, making possible to the expert to state both *hard* and *soft* constraints. As any convex constraint is allowed, it is possible to specify precise probability measures as well as vacuous beliefs (the specification of a single prior is also possible as it is a sub-case). Third, a single hyperparameter s must be elicited to capture the quality of the prior. Fourth, computations are efficient as all constraints are convex. Still, all these constraints define a set of distributions. Then we employ the maximum entropy principle, locally applied to each conditional distribution, to select one distribution from this set. Distributions of maximum entropy are conservative and tend to agree with frequencies [16]. So, the framework can be summarized as the following optimization problem:

$$\max_{\theta} - \sum_{ijk} \theta_{ijk} \log \theta_{ijk}, \quad (3)$$

subject to Equations (2), convex constraints $\mathcal{C}_p(\tau)$ and convex constraints $\mathcal{C}(\theta)$. This formulation can be polynomially solved by convex programming, as Equation (3) is the maximization of a concave function [8] subject to convex constraints. The resulting θ is our estimation.

Finally, we point out that Equations (2) tend the values of θ_{ijk} to be close to $\frac{n_{ijk}}{\sum_k n_{ijk}}$ (the frequencies of θ_{ijk} in the data set), while constraints on θ_{ijk} defined by the expert might (or might not) impose another different value. As we assume that hard constraints specified by the expert are correct, a penalty optimization variable is introduced in Equation (2) to guarantee that preference is given to the hard constraints defined by the expert and that the problem will not be infeasible (as long as expert's constraints are not already infeasible, in which case the expert should update her beliefs).

4 Experiments

We perform experiments using data sets with 10, 100 and 1000 samples. Three well–known networks are evaluated: Asia (Lauritzen and Spiegelhalter), Alarm (Beinlich et al.), and Insurance network (Binder et al.). For each network, we take one given parametrization as our true model and generate samples from it. Then Kullback–Leibler (KL) divergence is performed to measure the difference from distributions of estimated networks to distributions of true networks. Random linear (convex) constraints are generated from 1 to 5 parameters each. The constraints are created over the true network (so they are certainly correct) in number equal to the number of conditional distributions in the corresponding network. For each configuration, we work with 30 random sets of data and constraints. Averages of KL divergence are presented in Table 1. Columns have results of standard ML, Constrained ML and Constrained Maximum Entropy (CME).

Results indicate a strong decrease in the divergence when working with constraints (second and third columns of each block). Moreover, benefits are more significant with larger networks. We note that results with constraints using only 10 samples are better than results using 100 or even 1000 samples without constraints, which indicate a relevant decrease in the amount of data that would be necessary for achieving the same accuracy. The third column of each block shows results for the combination of constraints on priors and estimations using the maximum entropy idea. It is interesting to note the advantages when compared to the second column, because CML already uses constraints on estimations.

We also consider the problem of recognizing facial action units from real image data. Based on the Facial Action Coding System [6], facial behaviors can be decomposed into a set of Action Units (AUs). We work with a BN with 28 nodes to recognize 14 common occurring AUs (there are a hidden and a measurement node for each AU). The structure of the BN is learned as described in Tong et al. [13]. We define 42 simple linear constraints, mainly describing influences among AUs. The 8000 images from Cohn and Kanade's DFAT-504 database are used. Testing is performed over 20% of the data (not chosen for training). We consider training data sets with 100 and 1000 samples (as constraints are more relevant when insufficient data are available), chosen randomly from the training database. Results are shown in Table 2. CME obtains an overall recognition rate (percentage of correctly classified cases) of 93.1%, which is similar to current state-of-the-art results. For instance, Tong et at. [13] report 93.3%, Bartlett et al. [3] report 93.6%, and other methods have results with slight variance [12, 15].

5 Conclusion

This paper presents a framework for parameter learning when domain knowledge is available in the form

Network	Nodes	Distr	Dimension	10 samples			100 samples			1000 samples		
				ML	CML	CME	ML	CML	CME	ML	CML	CME
Asia	9	21	21	1.22	0.61	0.25	0.27	0.14	0.11	0.05	0.03	0.03
Alarm	37	243	509	3.26	1.80	0.61	2.51	1.19	0.47	1.12	0.58	0.26
Insurance	27	411	1008	3.62	1.56	0.63	2.24	0.92	0.44	0.96	0.40	0.20

Table 1. Average KL divergence using random samples and constraints.

Rate	100 samples		1000 samples	
	ML	CME	ML	CME
Positive	65.6%	75.7%	73.2%	83.9%
Negative	95.8%	97.7%	95.9%	97.0%

Table 2. Positive and negative rates for AU recognition.

of convex constraints. We have introduced a new idea based on the Imprecise Dirichlet Model and the maximum entropy criterion that is able to deal with constraints on priors and on estimations. The framework is fast and guarantees to find the global optimum solution. Through experiments with well-known networks, we show that both constraints on priors and estimations are important to improve parameter learning accuracy.

The main contribution of this work is to allow an expert to specify her knowledge using *hard* constraints on estimations and *soft* constraints on priors, with no restrictions on the format of constraints besides convexity. As far as we know, no previous methods were able to handle such general situation. The idea can also be embedded into an iterative procedure to treat incomplete data, similar to the Expectation-Maximization (EM) method. This discussion is left for the future, but we anticipate that the benefits are similar to those of complete data. We believe that the application of these ideas to real domains is promising and we intend to pursue that in a future work, as well as an investigation about the effect of wrong constraints.

Acknowledgments

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

[2] E. D. Andersen, B. Jensen, R. Sandvik, and U. Worsoe. The improvements in mosek version 5. Technical report, Mosek Aps, 2007.

[3] M. S. Bartlett, G. C. Littlewort, M. G. Frank, C. Lainscsek, I. Fasel, J. R. Movellan. Automatic Recognition of Facial Actions in Spontaneous Expressions. *Journal of Multimedia*, 1(6):22–35, 2006.

[4] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. MPS/SIAM Series, 2001.

[5] C. P. de Campos and F. G. Cozman. Belief updating and learning in semi-qualitative probabilistic networks. In *UAI*, p. 153–160, 2005.

[6] P. Ekman and W. V. Friesen. *Facial action coding system: A technique for the measurement of facial movement*. Consulting Psychologists Press, 1978.

[7] A. Feelders. A new parameter learning method for bayesian networks with qualitative influences. In *UAI*, p. 117–124, 2007.

[8] E. H. Lieb. Some convexity and subadditivity properties of entropy. *B. of the American Math. Soc.*, 81(1), 1975.

[9] R. S. Niculescu. *Exploiting Parameter Domain Knowledge for Learning in Bayesian Networks*. PhD thesis, Carnegie Mellon, 2005. CMU-CS-05-147.

[10] R. S. Niculescu, T. Mitchell, and B. Rao. Bayesian network learning with parameter constraints. *J. of Machine Learning Research*, 7(Jul):1357–1383, 2006.

[11] S. Renooij, L. C. van der Gaag, and S. Parsons. Context-specific sign-propagation in qualitative probabilistic networks. *Artif. Intell.*, 140(1-2):207–230, 2002.

[12] Y. Tian, T. Kanade, and J. Cohn. Recognizing action units for facial expression analysis. *IEEE Trans. on PAMI*, 23(2):97–115, 2001.

[13] Y. Tong, W. Liao, and Q. Ji. Facial action unit recognition by exploiting their dynamic and semantic relationships. *IEEE Trans. on PAMI*, p. 1683–1699, 2007.

[14] L. Utkin and T. Augustin. Decision making under incomplete data using the imprecise dirichlet model. *Int. J. of Approximate Reasoning*, 44(3):322–338, 2007.

[15] M. F. Valstar, I. Patras, and M. Pantic. Facial action unit detection using probabilistic actively learned support vector machines on tracked facial point data. In *CVPR, W. Vision for Human-Computer Interaction*, 2005.

[16] P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, London, 1991.

[17] P. Walley. Inferences from multinomial data: Learning about a bag of marbles. *J. Royal Statistical Society B*, 58(1):3–57, 1996.

[18] M. P. Wellman. Fundamental concepts of qualitative probabilistic networks. *Artif. Intell.* 44:257–303, 1990.