MAC6916 PROBABILISTIC GRAPHICAL MODELS
LECTURE 9: PARAMETER LEARNING

DENIS D. MAUÁ

1. Introduction

Thus far we have avoided the question "Where do probabilistic graphical models come from?", and we have taken for granted that a probabilistic model is available. However, obtaining a model that meets our requirements is far from trivial. One possibility is to rely on domain experts through a careful and painstaking process of *knowledge elicitation*. This includes training experts on probabilistic graphical modeling, calibrating experts' opinions and extracting and verifying knowledge. This process too often leads to disagreement among the experts and lack of reliability about the model. Nevertheless, in many domains where data is scarce, this is one of the main approaches to model construction. It is also part of the methodology of most scientific disciplines. Another possibility is to automatically derive the model using a data set of collected observations. It is this *machine learning* approach that we adopt here (so that we completely avoid the very rich field of human knowledge elicitation). To be fair, model construction is arguably never purely data-driven. A number of (often hidden) assumptions and constraints need to be made so that data can be turned into models. It is however domain agnostic: the machine learning approach can be almost automatically applied to any domain where data satisfying the requirements is available.

knowledge elicitation

Bayesian networks and Markov models (and also other probabilistic graphical models) have two parts: a qualitative part that encodes variable interactions and is represented by a graph, and a quantitative part that encodes the strength of interactions and is represented by real-valued functions. These parts are not independent, and learning of one part cannot be made without some assessment of the second part. Nevertheless, the distinction is very useful, as it allows us to decompose model construction in two computationally different problems: that of *structure learning*, where we infer a graph from data, and that of *parameter learning*, where we infer the numerical parameters given data and a *fixed* structure.

In this lecture, we look at the second task where the qualitative part of the model (i.e., the graph structure) is given, and we wish to learn the numerical parameters (e.g., the conditional probabilities associated with nodes of a Bayesian networks). We further divide the task in two scenarios according to the type of information available. In the *complete data* scenario, we assume that all relevant variables are *fully observable*, meaning that either we have a table of *complete instantiations* of variables we deem important, or we discard any variable/instantiation that contains a *missing value*. In the second scenario, we deal with *incomplete data*, that is, with

complete data

missing value

incomplete data

Date: April 26, 2016.

partial assignments that leave some of the variables unassigned. This might happen because some of the values could not be acquired during the data acquisition process (e.g., the respondent did not answer the question or the sensor failed), or because we assume that there are important quantities that cannot be directly measured and yet we wish to include them in the model. A simple example of the latter is *clustering*: we assume that data can be separated in groups of alike objects (as represented by their attributes), and we include a cluster indicator variable that labels each instance/object with its corresponding group.

Data-driven approaches to probabilistic model construction are the butter and bread of *statistical inference*. Most of the discussion here will be based in standard statistical principles such as maximum likelihood and consistency. These can be interpreted as *frequentist approches*. There are also *Bayesian approaches* that essentially cast model learning as inference in a Bayesian network with complex variables. For simplicity and uniformity, we use the notation and perspective of Bayesian inference, even when applying maximum likelihood, so that we can represent models of parameter learning as Bayesian networks.

2. Parameter Learning with Complete Data

2.1. **Bayesian networks.** Let us start with the simplest model imaginable: A Bayesian network with a single variable X taking on values 0/1 (see Figure 1). So suppose we have a list of observations $x_1, \dots, x_N$ of the values of X. This list can itself been seen as a *realization* of a list of variables $X_1, \dots, X_N$, each representing the desired outcome. In this simple case, there is only one possible structure and a single numerical parameter, namely, the probability $\theta = \mathbb{P}(X = 1)$.[1] A simple approach is to *guess* a value for θ and to estimate the *likelihood* that a model with that parameter would generate the observed data:[2]

$$\mathbb{P}(X_1 = x_1, \dots, X_N = x_N | \theta) .$$

maximum likelihood principle

The *Maximum Likelihood Principle* states that we should prefer a model that maximizes the quantity above. This criterion alone is insufficient for deriving a feasible strategy and additional assumptions have to be made. For instance, suppose we have data $X_1 = 1, X_2 = 1, X_3 = 1$; then it might appear that $\theta = 1$; however, unless additional assumptions are imposed, it might well be that all three observations are deterministically determined by some other (unobserved) variable. The most common assumption is that the observed variables are *exchangeable*, meaning that the order in which variables are arranged does not alter its probability (for any data set size). In other words, for any N and any permutation σ of the indices, it follows that

exchangeability

$$\mathbb{P}(X_{\sigma(1)} = x_{\sigma(1)}, \dots, X_{\sigma(N)} = x_{\sigma(N)} | \theta) = \mathbb{P}(X_1 = x_1, \dots, X_N = x_N | \theta) .$$

For example, in the thumbtack example we can assume that the conditions of the experiment remained unchanged so that the order of the outcomes is irrelevant. The famous de Finetti's representation theorem assures us that (under some very mild conditions) assuming exchangeability is the same as assuming *independent and identically distributed (i.i.d.) data* (the assumption that $X_1, \dots, X_N$ are indepen-

independent and identically distributed (i.i.d.) data

[1]Note that we also have to estimate $\mathbb{P}(X = 0)$, but this is defined by $\theta = 1 - \mathbb{P}(X = 0)$. The choice of $\mathbb{P}(X = 1)$ rather than $\mathbb{P}(X = 0)$ is arbitrary and unimportant.

[2]The notation $\mathbb{P}(X = x|\theta)$ suggests that we can specify a probability distribution $p(\theta)$; while this is not necessary, it is often desirable, as we will see later.

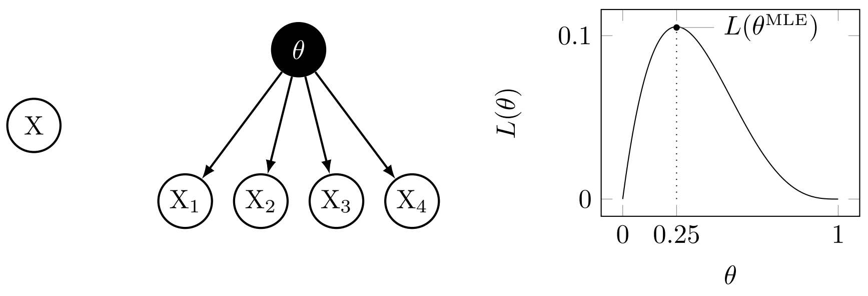

FIGURE 1. A Bayesian network with a single-variable (left), its corresponding parameter learning model (center), the data likelihood with $N = 4, N[X = 1] = 1$ (right).

dent given θ, and that each X_i is generated using the same distribution). Under the latter, we have that

$$\begin{aligned}\mathbb{P}(X_1 = x_1, \ldots, X_N = x_N|\theta) &= \prod_{j=1}^{N} \mathbb{P}(X = x_j|\theta) = \prod_{j=1}^{N} \theta^{x_j}(1-\theta)^{1-x_j} \\ &= \theta^{N[X=1]}(1-\theta)^{N[X=0]}\,,\end{aligned}$$

where $N[X = 1] = \sum_j x_j$ is the count of $X_j = 1$ and $N[X = 0] = N - N[X = 1]$. We can graphically represent these assumptions by the Bayesian network in Figure 1 (center). This network is known as the *parameter-learning Bayesian network*, augmented Bayesian network [2] and meta-Bayesian network [1, 3]. Note that for this simple case the parameter network has a simple naive Bayes structure.

parameter-learning Bayesian network

The likelihood is non-negative and bounded in $(0, 1)$, and zero at $\theta \in \{0, 1\}$. Recall that $f(x^*) \geq f(x)$ if and only if $\ln f(x^*) \geq \ln f(x)$ for any positive function. Since we are only interested in the maximizer of the likelihood (and assuming $0 < N[X = 1] < N$), we can instead maximize the *log-likelihood*:

log-likelihood

$$LL(\theta) = \ln \mathbb{P}(X_1 = x_1, \ldots, X_N = x_N|\theta) = N[X = 1] \ln \theta + N[X = 0] \ln(1 - \theta)\,.$$

The function above is strictly concave, bounded from above in $(0, 1)$ and vanishes at $\{0, 1\}$, therefore it has a single maximizer at a stationary point. We call that maximizer the *Maximum Likelihood Estimator* (MLE). Taking derivatives and setting to zero, we find that that MLE is

maximum likelihood estimator

$$\theta^{\text{MLE}} = \frac{N[X = 1]}{N}.$$

The plot in Figure 1 (right) depicts an example of the likelihood function with $N[X = 1] = 1$ and $N = 4$. The MLE is $\theta^{\text{MLE}} = 0.25$

Let us now turn to a slightly more complex model: a Bayesian network with two binary variables and an arc $X \to Y$. Figure 1 shows such a network (left), a Bayesian network representing the parameter learning model (center), and a more succinct version of the learning model in the form of a plates model (right). The desired parameters are the probabilities $\theta_X = \mathbb{P}(X = 1)$, $\theta_{Y|0} = \mathbb{P}(Y = 1|X = 0)$ and $\theta_{Y|1} = \mathbb{P}(Y = 1|X = 1)$. Let $\theta = (\theta_X, \theta_{Y|0}, \theta_{Y|1})$. The log-likelihood is given

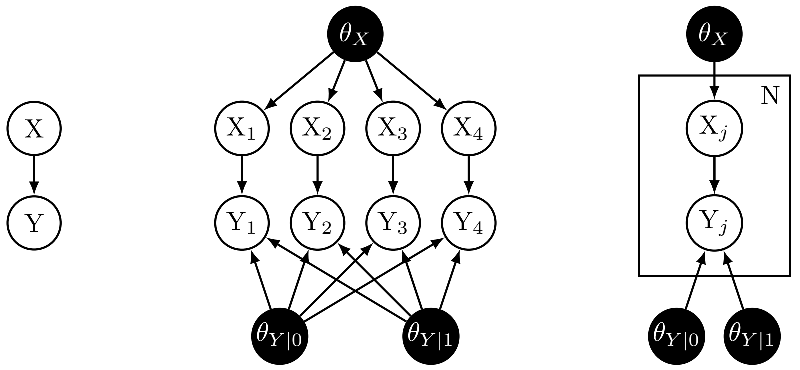

FIGURE 2. A Bayesian network to be learned (left), its corresponding parameter learning model (center), the equivalent plate model (right).

(under the same assumptions as before) by:

$$\begin{aligned}LL(\theta) &= \ln \mathbb{P}(X_1 = x_1, Y_1 = y_1, \dots, X_N = x_N, Y_N = y_N|\theta)\\ &= \sum_{j=1}^{N} \ln \mathbb{P}(X = x_j, Y = y_j|\theta) = \sum_{j=1}^{N}\Big(\ln \mathbb{P}(X = x_j|\theta_X) + \ln \mathbb{P}(Y = y_j|X = x_j, \theta_{Y|X})\Big)\\ &= \underbrace{\sum_{x\sim X} N[X = x] \ln \mathbb{P}(X = x|\theta_X)}_{LL(\theta_X)} + \underbrace{\sum_{y\sim Y} N[Y = y, X = 0] \ln \mathbb{P}(Y = y|X = 0, \theta_{Y|0})}_{LL(\theta_{Y|0})}\\ &\qquad + \underbrace{\sum_{y\sim Y} N[Y = y, X = 1] \ln \mathbb{P}(Y = y|X = 1, \theta_{Y|1})}_{LL(\theta_{Y|1})}\end{aligned}$$

Note that the overall log-likelihood decomposes as three functionally independent log-likelihood functions over single-variable models $LL(\theta_X)$, $LL(\theta_{Y|0})$ and $LL(\theta_{Y|1})$. Hence,

$$\begin{aligned}\theta^{\text{MLE}} &= (\theta_X^{\text{MLE}}, \theta_{Y|0}^{\text{MLE}}, \theta_{Y|1}^{\text{MLE}}) = \arg\max_{\theta} LL(\theta_X, \theta_{Y|0}, \theta_{Y|1})\\ &= \left(\arg\max_{\theta_X} LL(\theta_X), \arg\max_{\theta_{Y|0}} LL(\theta_{Y|0}), \arg\max_{\theta_{Y|1}} LL(\theta_{Y|1})\right) .\end{aligned}$$

Using the gradient we find that:

$$\theta_X^{\text{MLE}} = \frac{N[X = 1]}{N} \qquad \theta_{Y|0}^{\text{MLE}} = \frac{N[X = 0, Y = 1]}{N[X = 0]} \qquad \theta_{Y|1}^{\text{MLE}} = \frac{N[X = 1, Y = 1]}{N[X = 1]} ,$$

where $N[\eta]$ denotes the number of assignments where η is true, for example, $N[X = 0, Y = 1] = \sum_{j=1}^{N}(1 - x_j)y_j$.

It is easy to generalize this result to any arbitrary (discrete) Bayesian network. Let $\theta = (\theta_{X|\nu} : X \in \mathcal{V}, \nu \sim \text{pa}(X))$. The *complete data log-likelihood* of a Bayesian network over variable $\mathcal{V}$ is

complete data log-likelihood

$$LL(\theta) = \sum_{X\in\mathcal{V}} \sum_{x\sim X} \sum_{\nu\sim\text{pa}(X)} N[X = x, \text{pa}(X) = \nu] \ln \mathbb{P}(X = x|\text{pa}(X) = \nu, \theta_{X|\nu}) .$$

The MLE for each conditional probability is

$$\theta^{\text{MLE}}_{X|\nu}(x) = \frac{N[X = x, \text{pa}(X) = \nu]}{N[\text{pa}(X) = \nu]} .$$

Thus, parameter learning in Bayesian networks boils down to simple counting of relative frequencies for each configuration of variable and its parents. Note that the counts $N[X, \text{pa}(X)]$ can be stored in a potential and the probabilities can be obtained by a normalizing operation that normalizes function with respect to every restriction $\text{pa}(X) = \nu$.

The MLE for conditional probabilities is defined only when $N[\text{pa}(X) = \nu] > 0$. For small data sets, or for relatively large in-degree, it is often the case that $N[\text{pa}(X) = \nu]$ for some ν. A possible solution is to assume the *principle of indifference* (also called the principle of insufficient reason) that prescribes that we should assign equal probabilities to events whose likelihood are indistinguishable to us. This means assigning a uniform conditional distribution whenever $N[\text{pa}(X) = \nu] = 0$.[3] A more general form of the problem occurs when $N[\text{pa}(X) = \nu]$ is a small positive constant. In this scenario, the MLE is well-defined but very sensitive to local perturbations of the data (i.e., to change in a few instantiations), and thus not reliable (a few extra observations often change the implications of the model). To see this, consider the case where we estimate $\theta = \mathbb{P}(X = 1)$ for a binary variable X. Assume $N[X = 1]/N = 0.1$ and two scenarios: $N = 10$ and $N = 1000$. In the first scenario a single flip of an observation makes the estimate double its value (from 0.1 to 0.2) or become degenerate (i.e., $\theta = 0$). On the other hand, the estimate for the second scenario remains virtually unchanged even with a tenth of flips in the assignments. It is interesting to obtain estimators that are somehow *robust* to the sample size (meaning their variance is relatively small). This can be accomplished by introducing a inertia component known (in the case of discrete distributions) as *pseudo-counts*. The idea is essentially to insert virtual observations to every count, so that the estimator becomes

principle of indifference

robust estimator

pseudo-counts

$$\theta^{\text{BAYES}}_{X|\nu}(x) = \frac{N[X = x, \text{pa}(X) = \nu] + \alpha_{X|\nu}(x)}{\sum_{x'} N[X = x', \text{pa}(X) = \nu] + \alpha_{X|\nu}(x')} .$$

Each pseudo-count $\alpha_{X|\nu}(x)$ is a non-negative number (fractional counts are allowed) and indicate the amount of inertia. The sum of the pseudo-counts $\sum_x \alpha_{X|\nu}(x)$ indicates the *strength* of the belief in the background knowledge, and it is known as the *equivalent sample size*. When all hyper-parameters $\alpha_{X|\nu}(x)$ are set to unity, the corresponding prior distribution is a uniform on the space of probability distributions $\theta_{X|\nu}$ and the estimator is known as the *Laplace correction*.

equivalent sample size

Laplace correction

The pseudo-count approach solves both the large variance problem and the zero data problem, and it can be justified as the predictive Bayesian estimator. The Bayesian estimator for θ is obtained from the posterior probability:

$$p(\theta|\text{data}) = \frac{\mathbb{P}(\text{data}|\theta)p(\theta)}{\mathbb{P}(\text{data})} .$$

The term $\mathbb{P}(\text{data}|\theta)$ is the data likelihood, $p(\theta)$ is the prior and $\mathbb{P}(\text{data})$ is the *marginal data likelihood*. Note that $p(\theta)$ is actually a probability over a probability,

marginal data likelihood

[3]Note that this approach generates inconsistent probability models, with conditionals $\mathbb{P}(\alpha|\beta)$ defined for $\mathbb{P}(\beta) = 0$. Such models lie in the realm of the theory of full conditional probabilities [4].

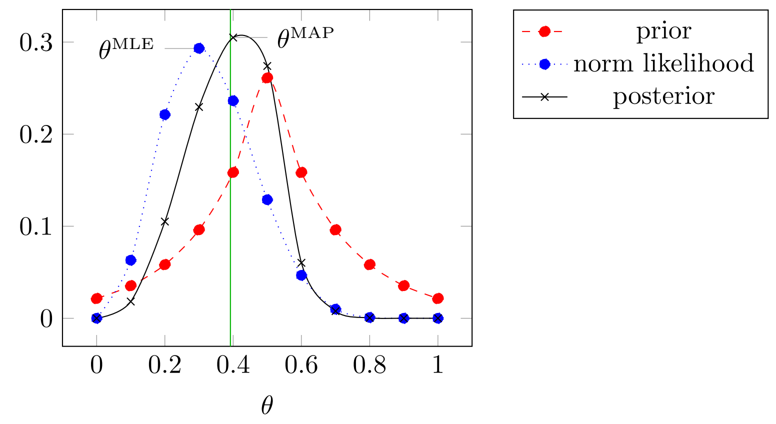

FIGURE 3. Prior, normalized likelihood and posterior distributions.

second-order probability

also called a *second-order probability*. The prior encodes background knowledge of the model (before observations are made), and the posterior is the combination of background knowledge and data likelihood (normalized by the marginal likelihood) as given by Bayes' rule. When X is a categorical variable and the prior is a Dirichlet distribution, the posterior is also a Dirichlet distribution. For the single-variable model we have that

$$\theta^{\text{BAYES}}(x) = \mathbb{P}(X = x|\text{data}) = \mathbb{E}(\theta(x)|\text{data}) = \int \theta \cdot p(\theta|\text{data}) d\theta = \frac{N[X = x] + \alpha_x}{N + \sum_{x'} \alpha_{x'}} \,.$$

The Bayes estimator integrates out all possible values of θ, which reduces the variance of the estimator. For example, consider again our scenario with $N = 10$ and $N[X = 1] = 3$, and assume that θ can only take values in $\{0, 0.1, 0.2, \ldots, 0.9, 1\}$ and that the prior is given by

$$p(\theta = \theta_i) \propto \exp\left(-5 * |0.5 - i|\right) \,.$$

This prior is peaked around 1/2 and decays exponentially fast as we move away from its peak. Figure 3 shows the prior distribution, the normalized likelihood (equivalent to a posterior with uniform prior), and the posterior distribution. The Bayesian estimator is

$$\theta^{\text{BAYES}} = \sum_{\theta_i} \theta_i \cdot p(\theta_i | N[X = 1] = 3, N = 10) = 0.392 \,.$$

This is shown as the vertical bar in the figure. The MLE estimate is $\theta^{\text{MLE}} = 0.3$.

When the data set is sufficiently large, the posterior probability $p(\theta|\text{data})$ will be concentrated around a small region. A suitable approximation is then given by the *maximum a posteriori* (MAP) estimator:

maximum a posteriori

$$\theta^{\text{MAP}} = \arg\max_{\theta} p(\theta|\text{data}) \,.$$

In the case of a categorical variable X with a Dirichlet prior this reduces to

$$\theta^{\text{MAP}}(x) = \frac{N[X = x] + \alpha_x - 1}{N - r + \sum_{x'} \alpha_{x'}} \,,$$

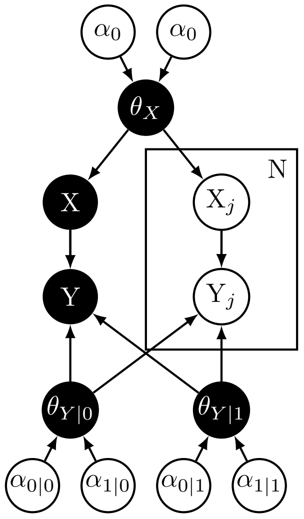

FIGURE 4. Plate model of the parameter learning Bayesian network for the network in Figure 2(left).

where $r = |\Omega_X|$. Note that the MAP estimator equals (for categorical variables) the Bayesian estimator up to a change of the pseudo counts (an unit increment). In general, the MAP estimator will be an approximation with larger variance that converges to the Bayesian estimator only in the limit. It is however often used in practice, as it is computationally less demanding.

This approach generalizes to any Bayesian network, but additional assumptions need to be made. The first assumption is that the prior distribution factorizes over each variable and configuration of its parents:

$$p(\theta) = \prod_{X \in \mathcal{V}} \prod_{\nu \sim \mathrm{pa}(X)} p(\theta_{X|\nu}) ,$$

where $\theta_{X|\nu} = \mathbb{P}(X|\mathrm{pa}(X) = \nu)$. This assumption is known as *prior parameter independence*. The second assumption it to assume that all variables are categorical and that $\mathbb{P}(\theta_{X|\nu})$ are Dirichlet distributions with hyperparameters $\alpha_{x|\nu}$:

prior parameter independence

$$p(\theta_{X|\nu}|\alpha_{X|\nu}) \propto \prod_{x \sim X} \theta_{X|\nu}(x)^{\alpha_{x|\nu}-1} .$$

The resulting parameter learning model is depicted in Figure 4 for the two-variable network in Figure 2. Note that if all X_j, Y_j are observed, the nodes θ_X and $\{\theta_{Y|0}, \theta_{Y|1}\}$ are d-separated. This is true in general:

$$p(\theta|\mathrm{data}) = \prod_{X \in \mathcal{V}} p(\theta_{X|\mathrm{pa}(X)}|\mathrm{data}) .$$

We say that an estimator is *consistent* if it converges to the true value. All the three estimators we have seen are consistent, that is, they recover the true distribution (assuming one exists) that generated the data as N goes to infinity. MLE converges faster (and with higher variance), while the Bayesian estimator converges the slowest (with smaller variance). In practice the Bayesian estimator often performs best, followed closely by the MAP estimator.

consistency

2.2. **Markov networks.** We have seen that parameter learning of Bayesian networks with complete data can be reduced to a problem of computing (smoothed) relative frequencies from the data set. This was related to the parameter independence, which allows us to estimate each conditional distribution independently. This was true even when the parameters were *integrated out*, as in the case of Bayesian estimators. Conceptually, the same approaches can be used to learn the parameters of a Markov network: maximum likelihood, Bayesian estimation, MAP. In practice, however, the unnormalized aspect of Markov networks makes the corresponding problem not decomposable, and hence much more difficult. To see this, consider a simple Markov network over potentials $\phi_1(X,Y)$ and $\phi_2(Y,Z)$. Its log-likelihood is

$$\begin{aligned} LL(\phi) &= \ln \mathbb{P}(\text{data}|\phi) \\ &= \underbrace{\sum_{x,y} N[X=y, Y=y] \ln \phi_1(x,y)}_{\approx LL(\phi_1)} + \underbrace{\sum_{y,z} N[Y=y, Z=z] \ln(y,z)}_{\approx LL(\phi_2)} - N \ln Z(\phi)\,, \end{aligned}$$

where

$$Z(\phi) = \sum_{x,y,z} \phi_1(x,y)\phi_2(y,z)\,.$$

As we can see from the expression above, maximizing the log-likelihood does not decompose into smaller problems due to the presence of the log-partition function $\ln Z(\phi)$ which *connects* both problems. For this particular network structure (whose domain graph is chordal), we can by-pass this problem by noticing that the model is Markov equivalent to any Bayesian network produced by a consistent orientation of the arcs, e.g.: $X \to Y$, $Y \to Z$. Since Markov equivalent models have the same log-likelihood, we can learn the corresponding Markov network by learning any consistent Bayesian network, e.g.:

$$\phi_1(X,Y) = \frac{N[X=x, Y=y]}{N[X=x]}\,, \qquad \phi_2(Y,Z) = \frac{N[Y=y, Z=z]}{N[Y=y]}\,.$$

This no longer holds on non-chordal graphs, and a Bayesian network-like estimation of the parameters leads to a suboptimal choice (under the maximum likelihood criteria).

log-linear parametrization

To make the notation simpler, we will use the following *log-linear parametrization* of a Markov network:

$$\mathbb{P}(\mathcal{V}|\theta) = \exp\left(\sum_k \theta_k f_k(\mathcal{V}) - \ln Z(\theta)\right)\,,$$

where θ_k are real values, and f_k are real-valued functions. Any positive Markov network can be parametrized in that way; in general the functions f_k will depend only on a subset of the variables (relative to cliques in the network). The log-likelihood is then

$$LL(\theta) = \sum_k \theta_k \left(\sum_{\nu \sim \mathcal{V}} N[\mathcal{V}=\nu] f_k(\nu)\right) - N \ln Z(\theta)\,.$$

Note that

$$\sum_{\nu \sim \mathcal{V}} \frac{N[\mathcal{V}=\nu]}{N} f_k(\nu)$$

is the empirical mean value of f_k in the data set, and does not depend on the parameters θ. Remarkably, $\ln Z(\theta)$ is convex in θ (it is the composition of an affine function and a sum of exponential [thus non-decreasing convex] compositions), hence $LL(\theta)$ is a concave function (since it is a sum of an affine function and a concave function). This guarantees the existence of an MLE at any stationary point, and the optimality of greedy search methods. First note that

$$\frac{\partial}{\partial\theta_k}\ln Z(\theta) = \frac{1}{Z(\theta)}\sum_{\nu}\exp\left(\sum_j \theta_j f_j(\nu)\right) f_k(\nu) = \sum_{\nu} f_k(\nu)\mathbb{P}(\nu|\theta)\,.$$

The stationary points of $LL(\theta)$ satisfy (for all k):

$$\frac{\partial}{\partial\theta_k}LL(\theta) = \frac{\partial}{\partial\theta_k}\sum_j \theta_j \sum_{\nu} N[\nu] f_j(\nu) - \frac{\partial}{\partial\theta_k} N \ln Z(\theta) = 0\,,$$

whence,

$$\sum_{\nu}\frac{N[\nu]}{N} f_k(\nu) = \sum_{\nu} f_k(\nu)\mathbb{P}(\nu|\theta)\,.$$

In words, the stationary points are the parameters that make the expected values of f_k equal to their empirical observed mean.

Even though the log-likelihood is a well-behaved function, it has no closed form solution and numerical iterative methods are required to solve the maximization for θ. A popular and relatively simple algorithm for solving a convex optimization (optimally) is *gradient ascent*, which consists of updates:

gradient ascent

$$\theta_k^{(t+1)} = \theta_k^{(t)} + \frac{\partial}{\partial\theta_k}LL(\theta^{(t)}) = \theta_k^{(t)} + \lambda_t \sum_{\nu} N[\nu] f_k(\nu) - \sum_{\nu} f_k(\nu)\mathbb{P}(\nu|\theta^{(t)})\,.$$

The parameter λ_t is a positive value denoted *learning rate* which regulates the convergence rate of the method. Too small values lead to slow convergence, while large values can lead to non-convergence. Typically, the search is initialized with a sufficiently large value, which is decreased at each iteration. Note that each iteration of the update above requires performing inference to compute the expected value of f_k (we can simultaneously compute all such values using e.g. sum-product or clique-tree algorithms).

learning rate

The computational difficulty with obtaining the MLE for Markov networks comes from the difficulty in computing the probability of a complete instantiation $\mathbb{P}(\nu)$, which can be performed in linear time in Bayesian networks. Thus, an alternative to the MLE estimate is to optimize a different function of the data which is based on some easy-to-compute probability. We have already seen that computing

$$\mathbb{P}(X|\mathcal{V}\setminus\{X\})$$

can be done efficiently if the Markov blanket of X is sufficiently small. This motivates the use of the *pseudo log-likelihood*:

pseudo log-likelihood

$$PLL(\theta) = \sum_{X\in\mathcal{V}}\sum_{\nu\sim\mathcal{V}} N[\nu]\ln\mathbb{P}(X=\nu[x]|\mathcal{V}\setminus\{X\}=\nu,\theta)\,.$$

The critical point here is that the pseudo log-likelihood can be computed without computing the partition function, and it is thus much more efficient. In fact, one can show that

$$\ln \mathbb{P}(X = \nu[x] | \mathcal{V} \setminus \{X\} = \nu, \theta) = \\ \ln \sum_{k:X \in \mathrm{sc}(f_k)} f_k(\nu) - \ln \left(\sum_{\nu'[\mathcal{V}\setminus\{X\}]=\nu[\mathcal{V}\setminus\{X\}]} \exp \left[\sum_{k:X\in\mathrm{sc}(f_k)} \theta_k f_k(\nu') \right] \right) .$$

Also,

$$\frac{\partial}{\partial \theta_k} PLL(\theta) = \\ \sum_{X:X\in\mathrm{sc}(f_k)} \sum_{\nu} N[\nu] \left(\frac{f_k(\nu)}{N} - \sum_{\nu'[\mathcal{V}\setminus\{X\}]=\nu[\mathcal{V}\setminus\{X\}]} f_k(\nu') \mathbb{P}(X = \nu'[x] | \mathcal{V} \setminus \{X\} = \nu', \theta) \right) .$$

The pseudo log-likelihood is a convex function, and its maximizers are consistent estimator; moreover, for sufficiently large samples, the maximizers of the pseudo log-likelihood coincide with the MLE estimates. This is not true with small data sets (even moderately large ones), and in this case the argument for PLL is somewhat weaker. PLL is particularly useful when the model is built so that most of the queries are similar to $\mathbb{P}(X|MB(X))$. This is the case for instance, in classification with few missing values, or when performing inference by Gibbs sampling.

3. Parameter Learning with Incomplete Data

3.1. **Bayesian networks.** Consider again the two-variable network in Figure 2, and assume we have data $y_1, \ldots, y_N$. That is, we observed only values for Y, and the values for X are missing. The observed data log-likelihood in this case is

$$LL(\theta) = \ln \prod_{j=1}^{N} \mathbb{P}(Y_j = y_j | \theta) = \sum_{j=1}^{N} \ln \left(\sum_{x_j} \mathbb{P}(Y_j = y_j | X_j = x_j) \mathbb{P}(X_j = x_j) \right) ,$$

which is equal to

$$N[Y=0] \cdot \ln \left([1-\theta_{Y|0}][1-\theta_X] + [1-\theta_{Y|1}]\theta_X \right) + N[Y=1] \cdot \ln \left(\theta_{Y|0}[1-\theta_X] + \theta_{Y|1}\theta_X \right) .$$

This is a much more complicated optimization, which has no closed formula (it is a non-convex function with multiple local maxima). More generally, assume that we have data $x_1, y_1, \ldots, x_N, y_N$, except that some values for x_j or y_j are missing (represented as $y_j = *$, $x_j = *$). For example (for now ignore the last column):

j	X	Y	part
1	0	1	D_1
2	1	1	D_1
3	*	1	D_2
4	0	*	D_3
5	0	1	D_1
6	*	*	D_4
7	0	0	D_1
8	1	0	D_1

The data log-likelihood is

$$
\begin{aligned}
LL(\theta) = & \sum_{j \in D_1} \ln \mathbb{P}(X_j = x_j, Y_j = y_j) + \sum_{j \in D_2} \ln \sum_{x_j} \mathbb{P}(X_j = x_j, Y_j = y_j) \\
& + \sum_{j \in D_3} \ln \sum_{y_j} \mathbb{P}(X_j = x_j, Y_j = y_j) + \sum_{j \in D_4} \ln \sum_{x_j, y_j} \mathbb{P}(X_j = x_j, Y_j = y_j)
\end{aligned}
$$

where

- D_1 are the pairs x_j, y_j where both are non-missing;
- D_2 are the pairs x_j, y_j where only x_j is missing;
- D_3 are the pairs x_j, y_j where only y_j is missing;
- D_4 are the pairs x_j, y_j where both are missing.

As a initial approximation let us estimate $\mathbb{P}(X = 1)$ using data sets D_1 and D_3:

$$\theta_X^{(0)} = 1/3 \,.$$

Similarly, we can estimate $\mathbb{P}(Y = 1|X)$ using D_1:

$$\theta_{Y|0}^{(0)} = 2/3 \qquad \theta_{Y|1}^{(0)} = 0.5 \,.$$

We can use these estimates to estimate the missing values. One alternative is to replace the missing values with the most probable values. For instance, we can choose $y_4 = \arg\max_y \mathbb{P}(Y = y|X = 0) = 1$ and

$$x_3 = \arg\max_x \mathbb{P}(X = x|Y = 1) \propto \mathbb{P}(X = x)P(Y = y|X = x) = 0 \,.$$

But this approach does not account for our uncertainty about the completion. An arguably better approach is to consider the expected value of each missing value. For instance, $\mathbb{E}(X_3|Y_3 = 1) = \mathbb{P}(X_3 = 1|Y_3 = 1) = 0.273$ and $\mathbb{E}(Y_4|X_4 = 0) = \mathbb{P}(Y_4 = 1|X_4 = 0) = 2/3$. We can interpret these values as *fractional counts*, and use them to compute the expected log-likelihood: fractional counts

$$
\begin{aligned}
ELL(\theta_X) &= \mathbb{E}(N[X = 1]) \ln(\theta_X) + \mathbb{E}(N[X = 0]) \ln(1 - \theta_X) \,, \\
ELL(\theta_{Y|0}) &= \mathbb{E}(N[Y = 1, X = 0]) \ln(\theta_{Y|0}) + \mathbb{E}(N[Y = 0, X = 0]) \ln(1 - \theta_{Y|0}) \,, \\
ELL(\theta_{Y|1}) &= \mathbb{E}(N[Y = 1, X = 1]) \ln(\theta_{Y|1}) + \mathbb{E}(N[Y = 0, X = 1]) \ln(1 - \theta_{Y|1}) \,, \\
ELL(\theta) &= ELL(\theta_X) + ELL(\theta_{Y|0}) + ELL(\theta_{Y|1}) \,.
\end{aligned}
$$

Note that the expression above is the same as in the case of complete data, except that the counts have been replaced with expected counts. Thus, we can find new estimates by maximizing the expected log-likelihood:

$$\theta^{(1)} = \arg\max_\theta ELL(\theta) \,.$$

This process can be repeated until convergence (which is guaranteed). This is known as the *Expectation-Maximization* algorithm [5], which alternates between an expectation step, where expected counts are computed, and a maximization step, where new estimates are produced by maximization of the expected log-likelihood. These two steps can be combined into a single update formula for arbitrary (discrete) Bayesian networks: Expectation-Maximization

$$\theta_{X|\nu}^{(t+1)}(x) = \frac{\sum_{j=1}^N \mathbb{P}(X = x, \text{pa}(X) = \nu|D_j, \theta^{(t)})}{\sum_{j=1}^N \mathbb{P}(\text{pa}(X) = \nu|D_j, \theta^{(t)})} \,,$$

where D_j is the assignments to observed variables in row j. To avoid poor local optimal, the algorithm is usually run several times with different initial estimates. Note that solving the equation above requires performing inference in the network. As with the complete data case, we can place priors over the parameters and maximize the posterior likelihood. Again, the result is a difficult non-convex optimization, which can be solved by EM. The update rule is then (for the MAP case):

$$\theta_{X|\nu}^{(t+1)}(x) = \frac{\sum_{j=1}^{N} \mathbb{P}(X = x, \text{pa}(X) = \nu | D_j, \theta^{(t)}) + \alpha_{x|\nu} - 1}{\sum_{j=1}^{N} \mathbb{P}(\text{pa}(X) = \nu | D_j, \theta^{(t)}) + \sum_{x'}(\alpha_{x|\nu} - 1)} ,$$

An alternative (and usually less efficient) approach to parameter learning, is to perform Gibbs sampling on the parameter-learning Bayesian network. This works with incomplete data, and even with continuous variables in the base model (as long as we can sample from its conditional distribution). However, this usually performs worse than the methods presented here. An improvement is to perform inference in the discrete part of the network conditional on the continuous parameter variables, and to analytically integrate out these variables, leading to a Rao-Blackwellized version of Gibbs sampling, which can be effective in practice, especially when the number of hidden variables is high.

3.2. **Markov networks.** The gradient ascent approach can be adapted for the case of incomplete data. The only change is that many inferences are required for the update of a single parameter at each iteration (as we computed the expected likelihood). An analogous version of Expectation-Maximization can also be used, where data is complete with expected counts (requiring inference) and then a new parameter is found (through e.g. gradient ascent). It is also possible to replace the last step with a maximization over the pseudo log-likelihood, thus making the procedure more efficient. Finally, Gibbs sampling can be applied, and it is often competitive with other approaches (especially if a Rao-Blackwellized version is used).

4. Reading

Sean Borman, The Expectation Maximization Algorithm: A short tutorial, 2004.

5. Exercises

No exercises this week.

6. Assignment

No assignment this week.

References

[1] A. Darwiche. Modeling and reasoning with Bayesian networks. Cambridge University Press, 2009.
[2] R.E. Neapolitan. Learning Bayesian Networks. Prentice-Hall, 2003.
[3] D. Koller and N. Friedman. Probabilistic Graphical Models MIT Press, 2009.
[4] P. Krauss Representation of conditional probability measures on Boolean algebras. Acta Mathematica Academiae Scientiarum Hungarica, volume 19, issues 3–4, pp. 229–241, 1968.
[5] A.P. Dempster, N.M. Laird and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B (Methodological), pp. 1–38, 1977.