Inference in Markov Blanket Networks

Reimar Hofmann

Siemens AG, Corporate Technology D-81730 München, Germany Reimar.Hofmann@mchp.siemens.de

Technical Report FKI-235-00 Technical University of Munich D-80290 Muenchen, Germany February 2000 (revised from a 1996 draft)

Abstract

Bayesian networks have been successfully used to model joint probabilities in many cases. When dealing with continuous variables and nonlinear relationships neural networks can be used to model conditional densities as part of a Bayesian network. However, doing inference can then be computationally expensive. Also, information is implicitly passed backwards through neural networks, i.e. from their output to the input. Used in this "inverse" mode neural networks often perform suboptimal. We suggest a different type of model called Markov blanket model (MBM). Here the neural networks are used in the forward direction only. This gives advantages in speed and guarantees to match the performance of the underlying neural network on complete data.

1 Introduction

Bayes nets (e.g. Heckerman (1995)) are models of the joint probability distribution of a set of variables $\{x_i\}_{i=1}^N$ of the form

$$p(x) = \prod_{i=1}^{N} p(x_i | \mathcal{P}_i).$$
(1)

where $\mathcal{P}_i \subseteq \{x_1, \ldots, x_{i-1}\}$ are the *parents*¹ of variable x_i .

¹Usually the smallest set will be used. \mathcal{P}_i is defined w. r. t. a given ordering of the variables.



Figure 1: Left: a Markov blanket model where neural networks are used to model the conditional probability $p^{M}(x_{i}|\mathcal{M}_{i})$. Right: the corresponding Markov network.

A Bayes net requires models of the conditional probability densities² $p(x_i|\mathcal{P}_i)$. Typically, they are implemented as tables (if variables are discrete) or as linear Gaussian models (if variables are continuous but the relations are linear). In Hofmann and Tresp (1996) we showed how Bayes nets can model very general nonlinear relationships between continuous variables by modeling the conditional densities using neural models such as multi-layer perceptrons, mixtures of experts or conditional Parzen windows.

One of the strengths of Bayes nets is that they allow inference from partial knowledge, i. e. states of unknown variables can be estimated from an arbitrary set of variables with known states. Although in general known to be an NP-hard problem, for many discrete or linear Bayes nets efficient believe update rules can be applied.

Correspondingly efficient update rules for propagation in Bayes nets of continuous variables with nonlinear dependencies are not available. A possible approach is to use Gibbs sampling. Gibbs sampling can be roughly described as follows: for all variables whose state is known, fix their states to the known values. For all unknown variables choose some initial states. Then pick a variable x_i which is not known and update its value following the probability distribution for x_i given all other variables

$$p(x_i|\{x_1,\ldots,x_N\}\setminus\{x_i\}) \propto \underbrace{p(x_i|\mathcal{P}_i)}_{\text{prior}} \underbrace{\prod_{x_i\in\mathcal{P}_j} p(x_j|\mathcal{P}_j)}_{\text{evidence}}.$$
(2)

Do this repeatedly for all unknown variables. Discard the first samples. Then, the samples which are generated are drawn from the probability distribution of the unknown variables given the known variables and can be used to calculate the expected value of any variable, estimate variances etc.

Gibbs sampling requires sampling from the univariate probability distribution in Equation 2 which is not straightforward in general since the conditional density does not have a convenient form. Therefore, sampling techniques such as *sampling-importance-resampling* have to be used. The idea is to generate samples according to a distribution for which this is easy, and then reject or accept them with a probability (depending on the sample) such that the distribution of the accepted samples is the desired distribution. In our case this typically produces many rejected samples and is therefore inefficient.

An alternative is sampling based on Markov blanket conditional density models (MBM): A

 $^{^{2}}$ For simplicity of notation we will only treat the continuous case throughout most of this paper. Results can be immediately transferred to discrete domains. We will use sloppy notation and discriminate probabilities by their arguments.

set \mathcal{M}_i with $p(x_i|\{x_1,\ldots,x_N\} \setminus x_i) = p(x_i|\mathcal{M}_i)$ is called a Markov blanket³ of x_i . (given a Bayes net, the Markov blanket of a variable consists of its parents, its children and its children's parents.). A MBM consists of Markov blankets \mathcal{M}_i and of conditional density models $p^{\mathcal{M}}(x_i|\mathcal{M}_i) \approx p(x_i|\mathcal{M}_i)$ for each variable x_i in the network (Figure 1). Sampling directly from these is usually much easier than sampling from the product in Equation 2. For example in in conditional Parzen models the conditional density is a mixture of Gaussians from which we can sample easily.

MBMs are also interesting if we are only interested in always predicting one particular variable, as in most neural network applications. Assuming that a signal-plus-noise model is a reasonably good model for the conditional density, we can train an ordinary neural network to predict the variable of interest. In addition, we train a model for each input variable predicting it from the remaining variables. This way we still use our well tested neural model for the complete data case, but we can now also handle missing inputs and do backward inference using Gibbs sampling.

There is another benefit to MBMs. It is well known that neural networks form very good forward models, that is they are very powerful to predict an output variable based on the input variables. In a MBM, if the variables in the Markov blanket are known we can use these very powerful models for prediction. Also during sampling, we only need to go in the forward direction through the model. If we sample in Bayes nets, we implicitly invert the forward models if we know the state of children and have to infer the states of the parents. Using neural models in this "inverse" mode typically is suboptimal.

Yet another benefit of MBM is that they are easier to learn from data. Learning the Markov blankets can be done independently for every variable and can not get stuck in local minima as can be the case learning the structure of Bayes nets if the ordering of variables in unknown.

The next section discusses some theoretical issues concerning MBMs. The following Section describes our experiments and results. Section 4 contains conclusions.

2 Theory

During this chapter we will always assume that all marginal and conditional probabilities and densities are strictly positive.

A MBM fully specifies the joint probability distribution. That is, the conditional densities $p(x_i|\{x_1, \ldots, x_N\} \setminus \{x_i\}) = p(x_i|\mathcal{M}_i)$ uniquely determine the joint density. This can be seen by noticing that these conditional densities are exactly what is needed to perform Gibbs sampling. A more direct proof can be found in Hasseln (1996).

Note, that reconstructing the joint density from a MBM is overdetermined. We can see this easily in the discrete case. Assume N binary variables, no independencies. In a Bayes net, we have $\sum_{i=1}^{N} 2^{i-1}$ free parameters. In a MBN, we would have $N \times 2^{N-1}$ free parameters. The conditional models can be *inconsistent* with respect to each other, i.e. there may be no joint density with conditionals as specified. This will be the case in general if the conditional models are determined from training data. Later in this paper we will examine the question what happens if we perform Gibbs sampling on inconsistent conditional models. Another issue is stability: consider the linear MBM for two continuous variables with $P(x_1|x_2) = G(x_1; ax_2, \sigma^2)$ and $P(x_2|x_1) = G(x_2; ax_1, \sigma^2)$. For |a| < 1 the corresponding joint density is Gaussian. With a approaching 1 the covariance between x_1 and x_2 approaches infinity, i.e. arbitrarily small changes in a can lead to major changes in the resulting joint density.

³A with respect to inclusion minimal Markov blanket is called Markov boundary.

In this sense Markov blanket sampling can be considered ill-posed. This raises the question under which conditions such instabilities can occur.

2.1 Relationship to Markov Networks

By specifying a Markov blanket \mathcal{M}_i for each variable x_i , a MBM specifies certain conditional independencies. We will show that a MBM can express all independencies that can be expressed by a *Markov network*⁴: Given a Markov network \mathcal{N} consider the MBM which defines, for each x_i , \mathcal{M}_i to be the set of neighbours of x_i in \mathcal{N} . The semantics of a Markov network guarantee that all independencies stated by the MBM also hold under \mathcal{N} . For the converse direction there is a theorem (Pearl (1988), Corollary 2, p. 98) stating that the Markov network of any strictly positive distribution can be constructed by connecting each variable to all Members of its Markov blanket⁵. Since this is the original Markov network this shows that all independencies in \mathcal{N} were also expressed by the MBM.

In (Hofmann and Tresp 1998) the relationship between MBMs and Markov networks is exploited to perform structural learning of Markov networks for nonlinear domains based on a MBM representation.

2.2 The Linear Case

Assume we build a MBM corresponding to a fully connected Markov network and assume further we use linear models for the conditional densities $p^M(x_i|X^{(i)}) = G(x_i; X^{(i)} \circ L_i, \sigma_i^2)$, where $G(.; c, \sigma^2)$ is our notation for an univariate normal distribution centered at c with variance σ_i^2 , X is the vector (x_1, \ldots, x_N) , $X^{(i)}$ is the vector $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)$ and $X^{(i)} \circ L_i$ is a scalar product. Given training data we can perform maximum likelihood (ML-) training on the parameters of the conditional density models. We obtain ML-estimates L_i^{ML} , $\sigma_i^{2,ML}$ $(i = 1 \ldots N)$.

Consider further a standard linear model $p^M(X) = G(X; 0, \Sigma)$ of the joint density (where $G(.; 0, \Sigma)$ is our notation for a multidimensional Gaussian centered at 0 with Covariance matrix Σ), and a model $p^M(X^{(i)}) = G(X^{(i)}; 0, \Sigma^{(i)})$ of the joint density of all variables except x_i , and assume we perform ML-training on the parameters to obtain estimators Σ^{ML} and $\Sigma^{(i),ML}$.

For every i the joint density can be factored according to

$$G(X;0,\Sigma) = G(X^{(i)};0,\Sigma^{(i)})G(x_i;X^{(i)} \circ L_i,\sigma_i^2),$$
(3)

and there is a one to one correspondence between the parameters $\{\Sigma\}$ one the one side and $\{\Sigma^{(i)}, L_i, \sigma_i^2\}$ on the other side of the equation. In other words both sides of the equation are just different parameterisations of the same model. We know from elementary statistics that the ML-model is independent of the parameterization used. Therefore all conditional models $G(x_i; X^{(i)} \circ L_i^{ML}, \sigma_i^{2,ML})$ with ML-parameters are consistent with the joint model $G((x_1, \ldots, x_N); 0, \Sigma^{ML})$ with ML-parameters.

This shows that in the fully connected linear case and with ML-training no inconsistencies occur within the Markov blanket model, and that the joint density represented by the Markov blanket model is identical to the standard linear model with ML-parameters. One

⁴Markov networks are graphical stochastical models, which represent conditional independencies in form of an undirected graph (see e.g. Pearl (1988) for a definition).

⁵The corollary is stated for Markov boundaries. Using the larger Markov blanket will lead to a non minimal Markov network, but still all independencies stated by the Markov network will be true.

can show that this holds not only for the fully connected structure making no independence assumptions, but also for any other structure of a Markov network *if all independence* and conditional independence statements made by the structure are true in the standard linear ML-model of the joint density. Similarly one can show that for MBMs in discrete domains using ML-learning of standard multinomial distributions no inconsistencies occur if all independence statements made by the MBM hold exactly for the empirical distribution of the data. In practice however, if one does not use a fully connected structure, the independence statements of the structure will rarely be fulfilled *exactly* by the distribution of the data.

2.3 Gibbs sampling from Inconsistent Conditional Densities

In the general case, inconsistencies can occur. Nevertheless, Gibbs sampling can be performed, and we will now attempt an analysis. We will assume discrete variables and do the analysis in the framework of finite state, discrete time, periodic, inhomogenous Markov chains. Let S be the finite state space, the state at time i be s_i and the row vector $p_i = (P(s_i))_{s_i \in S}$ be the probability distribution for s_i . Assume we start with s_0 drawn from the initial distribution p_0 , and update variables periodically in the order o_1, \ldots, o_G , where G is the number of unknown variables. The variable updated at time i is then $u_i := o_{(i \mod G)}$ and the update is according to $P^M(x_{u_i}|X^{(u_i)})$. This defines, for time i, the transition matrix⁶ $T_i^M := (P^M(s_i|s_{i-1}))_{s_{i-1},s_i}$ for positions where s_{i-1} and s_i differ in no other variable than u_i . All other entries are zero. Induction shows

$$p_i = p_0 \prod_{j=1}^i T_j^M$$

From $u_{i+G} = u_i$ we see $T_{i+G}^M = T_i^M$. Therefore, if we look only at times which are a multiple of G, the samples are drawn from a homogenous Markov chain with transition matrix U^M , and

$$p_{(iG)} = p_0 \cdot (U^M)^i$$

where $U^M = \prod_{j=1}^G T_{u_j}^M$. The theory of homogenous Markov chains tells us that a sufficient condition for U^M to be ergodic, i.e. for $p^{M,0} := \lim_{k \to \infty} (p_0 \cdot (U^M)^k)$ to exist and to be independent of p_0 , is that U^M has only nonzero entries. We ensure this by assuming that the transition probabilities $P^M(x_i|X^{(i)})$ are all nonzero. Clearly, $p^{M,0}U^M = p^{M,0}$.

If there are no inconsistencies then there exists a probability distribution q with $q T_i^M = q \forall i$. Then $q U^M = q$ follows immediately, and from the ergodicity of U^M also $q = p^{M,0}$. So q is the limit distribution at times which are a multiple of G. Now $q T_i^M = q \forall i$ implies that q is also the limit distribution for times which are not a multiple of G, i. e. q is the limit distribution for p_i as i goes towards infinity.

If the conditional probabilities are inconsistent there can be no limit distribution which is irrespective of the place in the update cycle. The limit distribution at position r in the update cycle then is

$$p^{M,r} := \lim_{k \to \infty} p_{(Gk+r)} = p^{M,0} \prod_{j=1}^{r} T^M_{u_j}$$
(4)

We are interested in the case where the *inconsistent* conditional probability models are approximations of the true *consistent* conditional probabilities. Assuming only small approximation errors, i.e. T_i^M close to transition matrices T_i based on the true conditional

⁶The "M" stands for "Model" since T_i^M is based on P^M .

probabilities, U^M is also close to U. We would like to find that then also $p^{M,r}$ is close to the true distribution p for all r. We know that $p^{M,0}$ and p are Eigenvectors to the Eigenvalue 1 of U^M and U respectively. Further, U^M and U are stochastic, strictly positive matrices. Sensitivity analysis for the Eigenvector problem gives a bound

$$\|p^{M} - p\|_{2} < \frac{K}{1 - \lambda_{2}} \|U^{M} - U\|_{2}$$
(5)

where K is independent of U^M and U, and λ_2 is the second largest Eigenvalue (the largest is always 1). The size of λ_2 also determines the convergence speed of the Gibbs sampler. So in cases where the Gibbs sampler converges fast, the final distribution will be insensitive to small approximation errors of the conditional models. This condition for stability can be transferred to bounded continuous domains, under conditions like continuity, by using discretization. An interesting question is whether the bound in Equation 5 is tight.

3 Experiments

In the experiment we will compare MBMs with a standard Parzen joint density model and with Bayes nets on real data.

Our Parzen joint density model was

$$p(x_1, \dots, x_N) = \sum_{k=1}^{D} G((x_1, \dots, x_N)); (x_1^k, \dots, x_N^k), (\sigma_1^2, \dots, \sigma_N^2))$$
(6)

where $\{x^j\}_{j=1}^D$ is the training set. The Gaussians are centered at (x_1^k, \ldots, x_N^k) which is the location of the k-th sample in the joint space. We use elliptic axis-parallel Gaussians with different variances $(\sigma_1^2, \ldots, \sigma_N^2)$ in each dimension. The variances were optimized based on the leave-one-out crossvalidation joint likelihood using a second order training algorithm.

In the MBM and in the Bayes net we used the corresponding Parzen conditional density estimators (compare Hofmann and Tresp (1995))

$$p^{M}(x_{i}|\mathcal{P}_{i}) = \frac{\sum_{k=1}^{D} G((x_{i},\mathcal{P}_{i}); (x_{i}^{k},\mathcal{P}_{i}^{k}), (\sigma_{i}^{2},\sigma_{\mathcal{P}_{i}}^{2}))}{\sum_{k=1}^{D} G(\mathcal{P}_{i};\mathcal{P}_{i}^{k},\sigma_{\mathcal{P}_{i}}^{2})},$$
(7)

The Gaussians in the nominator are centered at (x_i^k, \mathcal{P}_i^k) which is the location of the k-th sample in the joint input/output (or parent/child) space and the Gaussians in the denominator are centered at (\mathcal{P}_i^k) which is the location of the k-th sample in the input (or parent) space. $\sigma_{\mathcal{P}_i}^2$ is the vector of variances in the input space and $(\sigma_i^2, \sigma_{\mathcal{P}_i}^2)$ is the vector of variances in the joint input and output space. The Parzen conditional model is just the conditional density derived from the Parzen joint model, the difference is that the variances of the Parzen conditional density model are optimized based on the *conditional* leave-one-out likelihood.

Note that the Parzen joint model has the power to effectively remove unnecessary input variables by assigning them a large variance. This makes it unnecessary to explicitly determine the Markov blanket of each variable, implicit input pruning is performed by training the variances based on crossvalidation. In the case of Bayes nets structure learning is still required to find the arrow directions.

Our MBM was trained by independently optimizing the variances of the Parzen joint density models for each variable. The Bayes net was trained using the structure learning algorithm described in Hofmann and Tresp (1996).



Figure 2: Mean squared test set error predicting the housing price against number of missing inputs for Markov blanket model (solid), Bayes net (dashed) and joint Parzen (dashdotted).

We used the Boston housing data which is a data set with 506 samples. Each sample consists of the housing price and 14 variables which supposedly influence the housing price in a Boston neighbourhood. Two thirds of these samples were used for training, one third was used as test set. To measure performance we predicted the housing price on test data given all or a subset of the other variables. We used the mean squared prediction error to measure performance.

To predict the housing price using the Bayes net or the MBM we used Gibbs-sampling. For MBMs this is straightforward. For the Bayes net updating a variable x_i requires drawing samples from the product of prior and evidence in Equation 2. We used sampling-importance-resampling(see e.g. Bernardo, Smith, 1994, p. 350): Initial samples were generated according to the prior. The acceptance probability was the evidence divided by the upper bound⁷ b on the evidence. We choose

$$b(x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_N) = \prod_{\substack{x_c \in ch \, il \, dren(x_i)}} (\max_{k=1}^D G(x_c; x_c^k, \sigma_c^2)) \tag{8}$$

which can easily be seen to be an upper bound.

3.1 Results

Figure 2 shows the mean squared test set error in predicting the housing price. Input variables were taken away in fixed (arbitrary) order⁸.

The predictions of both the MBM and the Bayes net were computed as the average over a set of Gibbs samples. With the MBM we generated 2000 samples for every test data point. With the Bayes net we used the same amount of CPU time for each test data point as with the MBM. Depending on the rejection rate which can be very different for different data points this typically produced in the range of 20 to 1000 samples. Data points with high rejection rates generate less samples.

The MBM performs significantly better than the joint Parzen model over the whole range of missing inputs. For 0 to 2 missing inputs the MBM and the Bayes net show similar performance, for 3 missing inputs the Bayes net is worse than even the joint Parzen model. The likely explanation for this seems to be that many more samples would be required here to achieve competitive performance. These results are, however, preliminary. Results do vary from run to run and more experiments are scheduled. These will include evaluating the Bayes net with more than three missing variables and with more samples per data point.

⁷b is bound w.r.t. varying x_i . It may depend on all other variables.

⁸The order was: 'employment center', 'tax rate', 'crime rate', 'access to radial highways', 'pupil/teacher ratio'

4 Conclusions

We introduced Markov blanket models and gave several potential advantages over Bayesian networks. First experiments showed competitive results.

All experiments were based on Parzen windows. The only differences between the different types of models were the way the Parzen windows were trained and combined. This allows meaningful comparisons between the different models. The experimental comparison between MBMs and Bayes nets is not yet conclusive. More experiments will be conducted here. The next interesting step would then be to replace the Parzen windows by other types of neural networks more commonly used in the neural network community like Gaussian mixtures with learned center positions, or multilayer perceptron plus noise models.

References

Hasseln, H. (1996). An IPF procedure for conditionals: Maximum likelihood estimation for non-unique conditionally specified distributions. Instituto de Matematica Pura e Aplicada, Rio de Janeiro, Brasil. TR.

Heckerman, D. (1995). A tutorial on learning Bayesian networks. Microsoft Research, TR. MSR-TR-95-06.

Hofmann, R. and Tresp, V. (1996). Discovering structure in continuous variables using Bayesian networks. In: Neural Information Processing Systems 8, Proceedings of the 1995 Conference., 500-506.

Hofmann, R. and Tresp, V. (1998). Nonlinear Markov networks for continuous variables. In: Neural Information Processing Systems 10, Proceedings of the 1997 Conference, 521-527.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems. San Mateo, CA: Morgan Kaufmann.