Inference Networks for Graphical Models

Brooks Paige
Department of Engineering Science
University of Oxford
brooks@robots.ox.ac.uk

Frank Wood
Department of Engineering Science
University of Oxford
fwood@robots.ox.ac.uk

Abstract

We introduce a new approach for amortizing inference in directed graphical models. Inference in graphical models entails characterizing the joint distribution of latent random variables conditioned on observed random variables. We describe a procedure for constructing and learning a structured neural network which represents an inverse factorization of the graphical model, resulting in a conditional density estimator that takes as input particular values of the observed random variables, and returns an approximation the posterior distribution of the latent variables. This recognition model can be learned offline, independent from any particular dataset, prior to performing inference. The output from this network can be used either directly as a crude approximate estimator, or refined via importance sampling or sequential Monte Carlo to provide consistent estimates of posterior expectations and unbiased estimates of the marginal likelihood.

1 Introduction

Our goal is to greatly increase the space of models for which inference can become automatic. In graphical models with continuous and non-Gaussian conditional densities, or in graphical models whose factor graph representation contains loops, exact inference methods are not available and one must resort to variational approximations or sampling methods. As an alternative approach, we construct neural network models which themselves are trained to perform Bayesian inference directly, given a model specification.

An amortized inference procedure [8, 13] takes a model as its input, and generates an artifact which then can be leveraged for accelerating future inference tasks. In the context of Bayesian networks, this entails learning a function which maps from data — that is, settings of the observed nodes — to values or distributions over the latent variables.

In this work, we take a new approach to amortized inference in graphical models by considering it in the context of the sequential Monte Carlo (SMC) for graphical models algorithm introduced in Naesseth et al. [11]. The SMC algorithm targets a sequence of densities which is constructed by adding in factors from a factor graph one-at-a-time, culminating in a final density which is itself the entire graphical model. We structure our sequence of densities such that the observed variable nodes come first; then the latent variables are added one at a time. Critical to performance of sequential Monte Carlo schemes and importance sampling in general is the choice of proposal density. We introduce a modification of the masked autoregressive distribution estimator (MADE) [7] which is appropriate for real-valued multivariate conditional density estimation, and use this to learn expressive neural network factor representations from a family which is sufficiently flexible to recover (in some cases) the exact posterior density; when not exact, or when an approximate posterior is insufficient, the mismatch can be corrected for via SMC with a small number of particles.

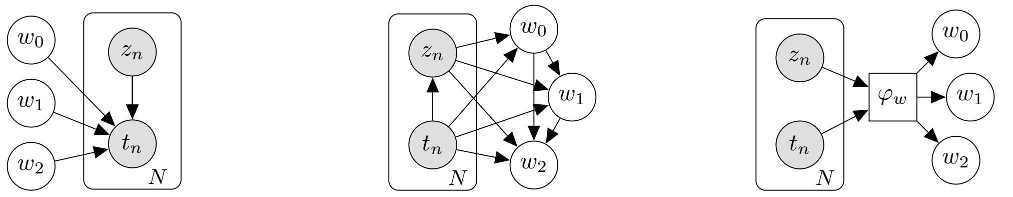

Figure 1: a non-conjugate regression model, as (left) a Bayes net representing a generative model for the data $\{t_n\}$; (middle) with dependency structure inverted, a generative model for the latent variables w_0, w_1, w_2; (right) showing the explicit neural network structure of the inverse conditional distribution $\tilde{p}(w_{0:2}|z_{1:N}, t_{1:N})$. Here we place a Laplace prior on each regression weight w_d, and have Student-t likelihoods $p(t_n|z_n, w_{0:2})$. New datasets $\{z_n, t_n\}_{n=1}^N$ can be input directly into the joint density estimator φ_w to estimate the posterior.

2 Approach

A directed graphical model, or Bayesian network [10, 12], defines a joint probability distribution and conditional independence structure via a directed acyclic graph. For each x_i in a set of random variables $x_1, \ldots, x_N$, the network structure specifies a conditional density $p_i(x_i|\textsc{pa}(x_i))$, where $\textsc{pa}(x_i)$ denotes the parent nodes of x_i. The joint distribution over N latent random variables $\mathbf{x}$ and M observed random variables $\mathbf{y}$ is defined as

$$p(\mathbf{x}, \mathbf{y}) \triangleq \prod_{i=1}^{N} p\left(x_i|\textsc{pa}(x_i)\right) \prod_{j=1}^{M} p\left(y_j|\textsc{pa}(y_j)\right); \tag{1}$$

the inference goal is to characterize the posterior distribution $\pi(\mathbf{x}) \equiv p(\mathbf{x}|\mathbf{y})$.

Our approach is two-fold. First, given a Bayesian network that acts as a generative model for our observed data $\mathbf{y}$ given latent variables $\mathbf{x}$, we construct a new Bayesian network which acts as a generative model for our latent $\mathbf{x}$, given observed data $\mathbf{y}$. This network is constructed such that the joint distribution defined by the original model $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y}|\mathbf{x})$ is identical to that of the new "inverse model", which we will refer to as $\tilde{p}(\mathbf{x}, \mathbf{y}) = \tilde{p}(\mathbf{y})\tilde{p}(\mathbf{x}|\mathbf{y})$, but with a different factorization [13].

Unfortunately, unlike the original forward model, the inverse model has conditional densities which we do not in general know how to normalize or sample from. However, were we to know the conditional densities comprising the inverse model $\tilde{p}(\mathbf{x}|\mathbf{y})$, then given a particular dataset $\mathbf{y}$ we could directly draw posterior samples simply by ancestral sampling from the inverse graphical model. Thus the second aspect is learning approximations for the conditionals $\tilde{p}(\mathbf{x}_i|\widetilde{\textsc{pa}}(\mathbf{x}_i))$, where $\widetilde{\textsc{pa}}(x_i)$ are parents of x_i in the inverse model. To do so we employ neural density estimators [1, 2, 7, 14], and design a procedure to train these "offline", in the sense that no real data is required.

As an example, consider the non-conjugate polynomial regression model shown in Figure 1, along with its inverse graphical model, and the resulting neural network structure. Note particularly that although the original graphical model which expressed $p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$ factorizes into products over y_n which are conditionally independent given $\mathbf{x}$, in the inverse model $\tilde{p}(\mathbf{x}|\mathbf{y})\tilde{p}(\mathbf{y})$ due to the explaining-away phenomenon all latent variables depend on all others.

2.1 Learning a family of importance sampling densities

Simple importance sampling in a Bayesian network performs inference by sampling $\mathbf{x}$ from some proposal density $q(\mathbf{x}|\cdot)$, and computing importance weights $w(\mathbf{x}) = p(\mathbf{x}, \mathbf{y})/q(\mathbf{x}|\cdot)$ which, for K samples of $\mathbf{x}$, yields a posterior approximation

$$\hat{p}(\mathbf{x}|\mathbf{y}) = \sum_{k=1}^{K} W_k \delta_{\mathbf{x}_k}(\mathbf{x}); \qquad W_k = \frac{w(\mathbf{x}_k)}{\sum_{j=1}^{K} w(\mathbf{x}_j)}. \tag{2}$$

The efficiency of the method depends crucially on the choice of proposal density. Previous work in adaptive importance sampling in a single-dataset setting (i.e., with fixed $\mathbf{y}$), both

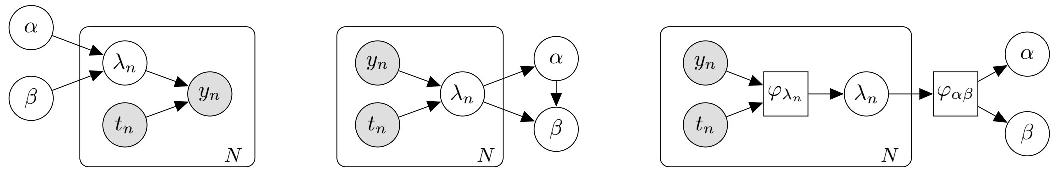

Figure 2: a hierarchical Bayesian model. (left) A generative model for the data $\{x_n\}$; (middle) with dependency structure inverted; (right) showing neural conditional density estimators. Each $y_n \sim \text{Poisson}(\lambda_n t_n)$, with $\lambda_n \sim \text{Gamma}(\alpha, \beta)$ and gamma priors on α, β. The learned factor φ_{λ_n} is replicated N times in the inverse model, allowing us to re-use the weights.

in the context of population Monte Carlo (PMC) [3] and sequential Monte Carlo [4, 5, 9], proposes a parametric family $q(\mathbf{x}|\lambda)$, where λ is a free parameter, and uses the reverse Kullback-Leibler (KL) divergence $D_{KL}(\pi||q_\lambda)$ as an objective function, choosing λ to minimize

$$D_{KL}(\pi||q_\lambda) = \int \pi(\mathbf{x}) \log \left[\frac{\pi(\mathbf{x})}{q(\mathbf{x}|\lambda)} \right] \mathrm{d}\mathbf{x}. \tag{3}$$

This KL divergence between the true posterior distribution $p(\mathbf{x}|\mathbf{y})$ and proposal distribution $q(\mathbf{x}|\lambda)$ is also known as the relative entropy criterion, and is a preferred objective function in situations in which the estimation goal construct a high-quality weighted sample representation, rather than to minimize the variance of a particular expectation [4].

In an amortized inference setting, instead of learning λ explicitly for a fixed value of $\mathbf{y}$, we learn a mapping from $\mathbf{y}$ to λ. More explicitly, if $\mathbf{y} \in \mathcal{Y}$ and $\lambda \in \vartheta$, then learning an explicit mapping $\varphi : \mathcal{Y} \to \vartheta$ allows performing approximate inference for $p(\mathbf{x}|\mathbf{y})$ with only the computational complexity of evaluating the deterministic function φ. The tradeoff is that the training of φ itself may be quite involved.

We thus generalize the adaptive importance sampling algorithms by learning a family of distributions $q(\mathbf{x}|\mathbf{y})$, parameterized by the observed data $\mathbf{y}$. Suppose that $\lambda = \varphi(\eta, \mathbf{y})$, where the function φ is parameterized by a set of higher-level parameters η. We would like a choice of η which performs well across all datasets $\mathbf{y}$. We can frame this as minimizing the expected value of Eq. 3 under $p(\mathbf{y})$, suggesting an objective function $\mathcal{J}(\eta)$ defined as

$$\begin{aligned}
\mathcal{J}(\eta) &= \int D_{KL}(\pi||q_\lambda) p(\mathbf{y}) \mathrm{d}\mathbf{y} && (4)\\
&= \int p(\mathbf{y}) \int p(\mathbf{x}|\mathbf{y}) \log \left[\frac{p(\mathbf{x}|\mathbf{y})}{q(\mathbf{x}|\varphi(\eta, \mathbf{y}))} \right] \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y} && (5)\\
&= \mathbb{E}_{p(\mathbf{x},\mathbf{y})} \left[-\log q(\mathbf{x}|\varphi(\eta, \mathbf{y})) \right] + \mathit{const.} && (6)
\end{aligned}$$

which has a gradient $\nabla_\eta \mathcal{J}(\eta) = \mathbb{E}_{p(\mathbf{x},\mathbf{y})} \left[-\nabla_\eta \log q(\mathbf{x}|\varphi(\eta, \mathbf{y})) \right]$.

Notice that these expectations are with respect to the tractable joint distribution $p(\mathbf{x}, \mathbf{y})$. We can thus fit η by stochastic gradient descent, estimating the expectation of the gradient $\nabla_\eta \mathcal{J}(\eta)$ by sampling synthetic full-data training examples $\{\mathbf{x}, \mathbf{y}\}$ from the original model. This procedure can be performed entirely offline — we require only to be able to sample from the joint distribution $p(\mathbf{x}, \mathbf{y})$ to generate candidate data points (effectively providing infinite training data). In any directed graphical model this can be achieved by ancestral sampling, where the variables $\mathbf{y}$ are treated as as-yet unobserved. Furthermore, we do not need need to be able to compute gradients of our model $p(\mathbf{x}, \mathbf{y})$ itself — we only need the gradients of our recognition model $q(\mathbf{x}|\varphi(\eta, \mathbf{y}))$, allowing use of any differentiable representation for q.

In hierarchical models such as the model for failure rates of power plant pumps [6] in Figure 2, conditional independence structure in an inverse model can be leveraged to break down $q(\mathbf{x}|\mathbf{y})$ into a product of smaller conditional densities, each of the form $q_i(x_i|\widetilde{\text{PA}}(x_i))$. We take advantage of this structure by defining more parameter-efficient representations of $q(\mathbf{x}|\cdot)$ that reuse replicated inverse conditional densities, and for more efficient inference via a sequential Monte Carlo algorithm.

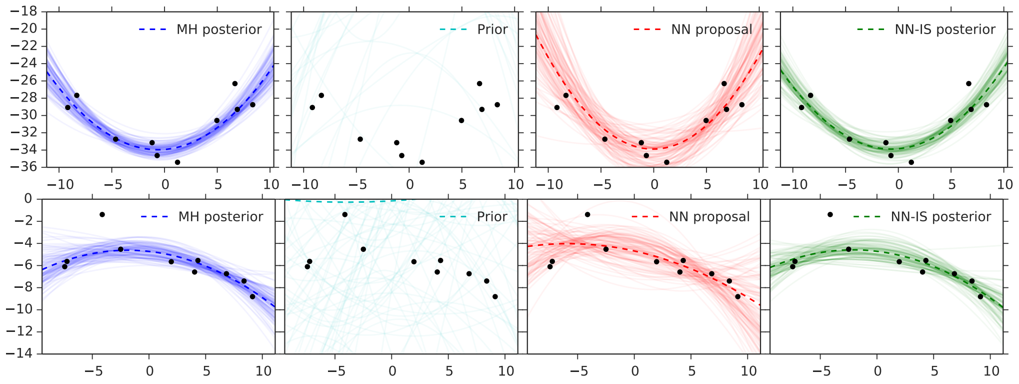

Figure 3: Representative output in the polynomial regression example. Plots show 100 samples each at 5% opacity, with the mean marked as a solid dashed line. All proposals use the same pre-trained neural network. The neural network proposal for the weights yields estimated polynomial curves close to the true posterior solution, albeit slightly more diffuse.

2.2 Conditional neural density estimation

We particularly wish to construct the inverse factorization $\tilde{p}(\mathbf{x}|\mathbf{y})$ (and our proposal model $q(\cdot)$) in such a way that we deal naturally with the presence of head-to-head nodes, in which one random variable may have a very large parent set. As in our example regression model, it is quite common to have generative models which factorize in the joint distribution, but have complex dependencies in the posterior. To address this we model such groups of random variables $\mathbf{x}_i$ jointly, with a neural conditional density estimator $q(\mathbf{x}_i|\varphi_i(\eta_i, \widetilde{\mathrm{PA}}(\mathbf{x}_i)))$ using a masked feed-forward network which extends MADE [7] for use on real-valued data, and for modeling conditional densities. Details are reserved for an extended version of this paper.

2.3 Sequential Monte Carlo for hierarchical models

In hierarchical models like that of Fig. 2 we can construct a sequential decomposition of the graphical model to run a sequential Monte Carlo algorithm. In a simple importance sampling algorithm, at inference time we propose all latent variables simultaneously by sampling from $q(\mathbf{x}|\mathbf{y})$, followed by a final importance weighting step. By contrast, an SMC algorithm for graphical models [11] sweeps through a sequence of target densities defined on successively larger sets of latent variables, following the ordering defined by the inverse model $\tilde{p}(\mathbf{x}|\mathbf{y})$. In this context the learned approximations to each $\tilde{p}(\mathbf{x}_i|\widetilde{\mathrm{PA}}(\mathbf{x}_i))$ aim to recover a sequence of optimal incremental proposal densities.

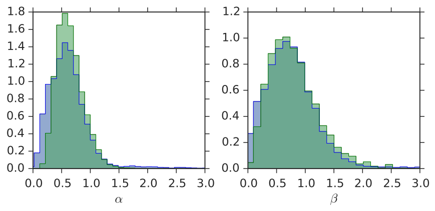

Figure 4: Representative output from the Poisson model: proposals (shown in blue) and posteriors (in green) for both α and β, estimated at the end of an SMC run.

3 Discussion

Representative results for the regression model are shown in Figure 3; learned distributions for the Poisson example are shown in Figure 4, tested on the actual power pump failure data. We recover estimators which closely resemble the true posterior, despite having never seen the data itself during training.

Though we presented this primarily as a manner by which we compile away application-time inference costs, from another perspective the neural network itself can be seen as the desired artifact: we provide instead a graphical-model-regularized neural network training algorithm that produces neural networks which compute interpretable structural representations that account for uncertainty in a way that mimics inference in a structured graphical model.

We thank Tom Jin and Jan-Willem van de Meent for their helpful discussions and comments.

References

[1] Yoshua Bengio and Samy Bengio. Modeling high-dimensional discrete data with multi-layer neural networks. In *Advances in Neural Information Processing Systems*, volume 99, pages 400–406, 1999.

[2] Christopher M Bishop. Mixture density networks. 1994.

[3] Olivier Cappé, Randal Douc, Arnaud Guillin, Jean-Michel Marin, and Christian P Robert. Adaptive importance sampling in general mixture classes. *Statistics and Computing*, 18(4):447–459, 2008.

[4] Julien Cornebise, Éric Moulines, and Jimmy Olsson. Adaptive methods for sequential importance sampling with application to state space models. *Statistics and Computing*, 18:461–480, 2008.

[5] Julien Cornebise, Éric Moulines, and Jimmy Olsson. Adaptive sequential Monte Carlo by means of mixture of experts. *Statistics and Computing*, 24:317–337, 2014.

[6] Edward I George, UE Makov, and AFM Smith. Conjugate likelihood distributions. *Scandinavian Journal of Statistics*, pages 147–156, 1993.

[7] Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. MADE: masked autoencoder for distribution estimation. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 881–889, 2015.

[8] Samuel J Gershman and Noah D Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the Thirty-Sixth Annual Conference of the Cognitive Science Society*, 2014.

[9] Shixiang Gu, Zoubin Ghahramani, and Richard E Turner. Neural adaptive sequential Monte Carlo. In *Advances in Neural Information Processing Systems 28*, 2015.

[10] David Heckerman, Dan Geiger, and David M Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243, 1995.

[11] Christian A. Naesseth, Fredrik Lindsten, and Thomas B. Schön. Sequential Monte Carlo for Graphical Models. In *Advances in Neural Information Processing Systems 27*. 2014.

[12] Judea Pearl and Stuart Russell. *Bayesian networks*. Computer Science Department, University of California, 1998.

[13] Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. Learning stochastic inverses. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3048–3056. 2013.

[14] Benigno Uria, Iain Murray, and Hugo Larochelle. RNADE: The real-valued neural autoregressive density-estimator. In *Advances in Neural Information Processing Systems*, pages 2175–2183, 2013.