Bayesian structure learning for Markov Random Fields with a spike and slab prior

Chen, Y.; Welling, M.

Published in:

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Bayesian Structure Learning for Markov Random Fields with a Spike and Slab Prior

Yutian Chen
Department of Computer Science
University of California, Irvine
Irvine, CA 92697

Max Welling
Department of Computer Science
University of California, Irvine
Irvine, CA 92697

Abstract

In recent years a number of methods have been developed for automatically learning the (sparse) connectivity structure of Markov Random Fields. These methods are mostly based on $L_1$-regularized optimization which has a number of disadvantages such as the inability to assess model uncertainty and expensive cross-validation to find the optimal regularization parameter. Moreover, the model’s predictive performance may degrade dramatically with a suboptimal value of the regularization parameter (which is sometimes desirable to induce sparseness). We propose a fully Bayesian approach based on a “spike and slab” prior (similar to $L_0$ regularization) that does not suffer from these shortcomings. We develop an approximate MCMC method combining Langevin dynamics and reversible jump MCMC to conduct inference in this model. Experiments show that the proposed model learns a good combination of the structure and parameter values without the need for separate hyper-parameter tuning. Moreover, the model’s predictive performance is much more robust than $L_1$-based methods with hyper-parameter settings that induce highly sparse model structures.

1 Introduction

Undirected probabilistic graphical models, also known as Markov Random Fields (MRFs), have been widely used in a large variety of domains including computer vision (Li, 2009), natural language processing (Sha and Pereira, 2003), and social networks (Robins et al., 2007). The structure of the model is defined through a set of features defined on subsets of random variables. Automated methods to select relevant features are becoming increasingly important in a time where the proliferation of sensors make it possible to measure a multitude of data-attributes. There is also an increasing interest in sparse model structures because they help against overfitting and are computationally more tractable than dense model structures.

In this paper we focus on a particular type of MRF, called a log-linear model, where structure learning or feature selection is integrated with parameter estimation. Although structure learning has been extensively studied for directed graphical models, it is typically more difficult for undirected models due to the intractable normalization term of the probability distribution, known as the partition function. Traditional algorithms apply only to restricted types of structures with low tree-width (Andrew and Gao, 2007; Tsuruoka et al., 2009; Hu et al., 2009) or special models such as Gaussian graphical models (Jones et al., 2005) so that accurate inference can be conducted efficiently.

For an arbitrary structure, various methods have been proposed in the literature, generally categorized into two approaches. One approach is based on separate tests on an edge or the neighbourhood of a node so that there is no need to compute the joint distribution (Wainwright et al., 2007; Bresler et al., 2008; Ravikumar et al., 2010). The other approach is based on maximum likelihood estimation (MLE) with a sparsity inducing criterion. These methods require approximate inference algorithms in order to estimate the log-likelihood such as Gibbs sampling (Della Pietra et al., 1997), loopy belief propagation (Lee et al., 2006; Zhu et al., 2010), or pseudo-likelihood (Howling and Tibshirani, 2009). A popular choice of such a criterion is $L_1$ regularization (Riezler and Vasserman, 2004; Dudik et al., 2004) which enjoys several good properties such as a convex objective function and a consistency guarantee. However, $L_1$-regularized MLE is usually sensitive to the choice the regularization strength, and these optimization-based methods cannot provide a credible interval for the learned structure. Also, in order to learn a sparse structure, a strong penalty has to be imposed on all the edges which usually results in suboptimal parameter values.

We will follow a third approach to MRF structure learning in a fully Bayesian framework which has not been ex-
explored yet. The Bayesian approach considers the structure of a graphical model as random. Inference in a Bayesian model provides inherent regularization, and offers a fully probabilistic characterization of the underlying structure. It was shown in Park and Casella (2008) that Bayesian models with a Gaussian or Laplace prior distribution (corresponding to $L_2$ or $L_1$ regularization) do not exhibit sparse structure. Mohamed et al. (2011) proposes to use a “spike and slab” prior for learning directed graphical models which corresponds to the ideal $L_0$ regularization. This model exhibits better robustness against over-fitting than the related $L_1$ approaches. Unlike the Laplace/Gaussian prior, the posterior distribution over parameters for a “spike and slab” prior is no longer guaranteed to be unimodal. However, approximate inference methods have been successfully applied in the context of directed models using MCMC (Mohamed et al., 2011) and expectation propagation (Hernandez-Lobato et al., 2010).

Unfortunately, Bayesian inference for MRFs is much harder than for directed networks due to the partition function. This feature renders even MCMC sampling intractable which caused some people to dub these problems “double intractability” (Murray et al., 2006). Nevertheless, variational methods (Parise and Welling, 2006; Qi et al., 2005) and MCMC methods (Murray and Ghahramani, 2004) have been successfully explored for approximating inference when the model structure is fixed.

We propose a Bayesian structure learning method with a spike and slab prior for MRFs and devise an approximate MCMC method to draw samples of both the model structure and the model parameters by combining a modified Langevin sampler with a reversible jump MCMC method. Experiments show that the posterior distribution estimated by our inference method matches the actual distribution very well. Moreover, our method offers better robustness to both under-fitting and over-fitting than $L_1$-regularized methods. A related but different application of the spike and slab distribution in MRFs is shown in Courville et al. (2011) for modelling hidden random variables.

This paper is organized as follows: we first introduce a hierarchical Bayesian model for MRF structure learning in section 2 and then describe an approximate MCMC method in section 3, 4, 5 to draw samples for the model parameters, structure, and other hyper-parameters respectively. Experiments are conducted in section 6 on two simulated data sets and a real-world dataset, followed by a discussion section.

## 2 Learning the Structure of MRFs as Bayesian Inference

The probability distribution of a MRF is defined by a set of potential functions. Consider a widely used family of MRFs with log-linear parametrization:

$$P(x|\theta) = \frac{1}{Z(\theta)} \exp \left( \sum_\alpha \theta_\alpha f_\alpha(x_\alpha) \right)$$

(1)

where each potential function is defined as the exponential of the product between a feature function $f_\alpha$ of a set of variables $x_\alpha$ and an associated parameter $\theta_\alpha$. $Z$ is called the partition function. All the variables in the scope of a potential function form a clique in their graphical representation. When a parameter $\theta_\alpha$ has a value of zero, we could equivalently remove feature $f_\alpha$ and all the edges between variables in $x_\alpha$ (if these variables are not also in the scope of other features) without changing the distribution of $x$. Therefore, by learning the parameters of this MRF model we can simultaneously learn the structure of a model if we allow some parameters to go to zero.

The Bayesian learning approach to graphical models considers parameters as a random variable subject to a prior. Given observed data, we can infer the posterior distribution of the parameters and their connectivity structure. Two commonly used priors, the Laplace and the Gaussian distribution, correspond to the $L_1$ and $L_2$ penalties respectively in the associated optimization-based approach. Although a model learned by $L_1$-penalized MLE is able to obtain a sparse structure, the full Bayesian treatment usually results in a fully connected model with many weak edges as observed in Park and Casella (2008), without special approximate assumptions like the ones in Lin and Lee (2006). We propose to use the “spike and slab” prior to learn a sparse structure for MRFs in a fully Bayesian approach. The spike and slab prior (Mitchell and Beauchamp, 1988; Ishwaran and Rao, 2005) is a mixture distribution which consists of a point mass at zero (spike) and a widely spread distribution (slab):

$$P(\theta_\alpha) = (1 - p_0) \delta(\theta_\alpha) + p_0 \mathcal{N}(\theta_\alpha; 0, \sigma_0^2)$$

(2)

where $p_0 \in [0, 1]$, $\delta$ is the Dirac delta function, and $\sigma_0$ is usually large enough to be uninformative. The spike component controls the sparsity of the structure in the posterior distribution while the slab component usually applies a mild shrinkage effect on the parameters of the existing edges even in a highly sparse model. This type of selective shrinkage is different from the global shrinkage imposed by $L_1/L_2$ regularization, and enjoys benefits in parameter estimation as demonstrated in the experiment section.

The Bayesian MRF with the spike and slab prior is formulated as follows:

$$P(x|\theta) = \frac{1}{Z(\theta)} \exp \left( \sum_\alpha \theta_\alpha f_\alpha(x_\alpha) \right)$$

$$\theta_\alpha = Y_\alpha A_\alpha$$

$$Y_\alpha \sim \text{Bern}(p_0) \quad p_0 \sim \text{Beta}(a, b)$$

$$A_\alpha \sim \mathcal{N}(0, \sigma_0^2) \quad \sigma_0^{-2} \sim \Gamma(c, d)$$

(3)
where \( x \) is a set of state variables and \( a, b, c, \) and \( d \) are hyper-parameters. In the experiments we will use pairwise features in which case \( \alpha = (i,j) \) plus bias terms given by \( \sum_i \theta_i f_i(x_i) \) in the expression for the log-probability. We will use a normal prior \( \theta_i \sim \mathcal{N}(0, \sigma_0^2) \) for these bias terms. \( \sigma_0 \) is chosen to be large enough to act as an uninformative prior. \( Y_\alpha \) is a binary random variable representing the existence of the edges in the clique \( x_\alpha \), and \( A_\alpha \) is the actual value of the parameter \( \theta_\alpha \) when the edges are instantiated. It is easy to observe that given \( p_0 \) and \( \sigma_0 \), \( \sigma_\alpha \) has the same distribution as in equation 2. We use a hierarchical model for \( \theta \) so that the inference will be insensitive to the choice of the hyper-parameters. In fact, experiments show that with a simple setting of the hyper-parameters, proper values of the sparsity parameter \( p_0 \) and the variance \( \sigma_0 \) are learned automatically by our model for all the data sets without the necessity of cross-validation.

Unlike the optimization-based methods which estimate a single structure, the Bayesian approach expresses uncertainty about the existence of edges through its posterior distribution, even when allowed a few steps at every iteration. In a MRF, the gradient term \( g \) involves computing an expectation over exponentially many states as

\[
g_\alpha(\theta) = \sum_{m=1}^N f_\alpha(x^{(m)}_\alpha) - N \mathbb{E}_{P(x|\theta)} f_\alpha(x_\alpha) - \frac{\theta_\alpha}{\sigma_0^2} \tag{5}\]

The expectation is estimated by a set of state samples \( \{x^{(s)}\} \) in the “brief Langevin” algorithm of Murray and Ghahramani (2004), where these samples are drawn by running a few steps of Gibbs sampling initialized from a subset of the training data. We adopt the “brief Langevin” algorithm with three modifications for faster mixing.

### 3.1 Persistent Gibbs Sampling

We maintain a set of persistent Markov chains for the state samples \( \{x^{(s)}\} \) by initializing Gibbs sampling at the last states of the previous iteration instead of the data. This is motivated by the persistent contrastive divergence algorithm of Tieleman (2008). When \( \theta \) changes slowly enough, the Gibbs sampler will approximately sample from the stationary distribution, even when allowed a few steps at every iteration.

We omit all the other random variables that \( \theta \) is conditioned on in this section for ease of notation.
3.2 Preconditioning

When the posterior distribution of \( \{ \theta_a \} \) has different scales along different variables, the original LMC with a common step size for all \( \theta_a \)'s will traverse the parameter space slowly. We adopt a preconditioning matrix \( C \) to speed up the mixing, where \( C \) satisfies \( CC^T = H \) with \( H \) is the Hessian matrix of \( \log P(\theta_{MAP}|D) \), computed as:

\[
H(\theta_{MAP}) = \text{cov}_{P(x|\theta_{MAP})} f(x) + \sigma_0^{-2}
\]

This is reminiscent of the observed Fisher information matrix in [Girolami and Calderhead, 2011] except that we use the MAP estimate with the prior. We approximate \( H(\theta_{MAP}) \) by averaging over \( H(\theta_t) \) during a burn-in period and estimate \( \text{cov}_{P(x|\theta)} f \) with the set of state samples from the persistent Markov chains. The adoption of a preconditioning matrix also helps us pick a common step size parameter \( \varepsilon \) suitable for different training sets.

3.3 Partial Momentum Refreshment

The momentum term \( p \) in the leapfrog step represents the update direction of the parameter. Langevin dynamics is known to explore the parameter space through inefficient random walk behavior because it draws an independent sample for \( p \) at every iteration. We can achieve a better mixing rate with the partial momentum refreshment method proposed in [Horowitz, 1991]. When \( p \) is updated at every step by:

\[
p_t \leftarrow \alpha p_t + \beta v_t
\]

where \( v_t \sim N(0, I) \), and \( \alpha, \beta \) satisfy \( \alpha^2 + \beta^2 = 1 \), the momentum is partially preserved from the previous iteration and thereby suppresses the random-walk behavior in a similar fashion as HMC with multiple leapfrog steps.

\( \alpha \) controls how much momentum to be carried over from the previous iteration. With a large value of \( \alpha \), LMC reduces the auto-correlation between samples significantly relative to LMC without partial momentum refreshment. The improved mixing rate is illustrated in Figure 2. We also show that the mean and standard deviation of the posterior distribution does not change. However, caution should be exercised especially when the step size \( \eta \) is large because a value of \( \alpha \) that is too large would increase the error in the update equation which we do not correct with a Metropolis-Hastings step because that is intractable.

4 Sampling Edges by Reversible Jump MCMC

Langevin dynamics handles the continuous change in the parameter value \( A_a \). As for discrete changes in the model structure, \( Y_a \), we propose an approximate reversible jump MCMC (RJMCMC) step ([Green, 1995]) to sample \( Y_a \) and \( A_a \) jointly from the conditional distribution \( P(A, Y|p_0, \sigma_0, D) \). The proposed Markov chain adds/deletes one clique (or simply one edge when \( \alpha = 1 \)) at a time. When an edge does not exist, i.e., \( Y_a = 0 \), the variable \( A_a \) can be excluded from the model, and therefore we consider the jump between a full model with \( Y_a \neq 0 \), \( A_a = a \) and a degenerate model with \( Y_a = 0 \).

The proposed RJMCMC is as follows: when \( Y_a = 0 \), propose adding an edge with probability \( P_{add} \) and sample \( A_a = a \) from a proposal distribution \( q(A) \) with support on \([-\Delta_a, \Delta_a] \); when \( Y_a = 1 \) and \( |A_a| \leq \Delta_a \), propose deleting an edge with \( P_{del} \). The reason of restricting the proposed move within \([-\Delta_a, \Delta_a] \) will be explained later. It is easy to see that the Jacobian is 1. The jump is then accepted by the Metropolis-Hastings algorithm with a probability:

\[
Q_{add} = \min\{1, Q^*(a)\}, \quad Q_{del} = \min\{1, 1/Q^*(a)\}
\]

\[
Q^*(a) = \exp\left( a \sum_m f_a(x^{(m)}_a) \right) \left( \frac{Z(Y_a = 0)}{Z(Y_a = 1, A_a = a)} \right)^N 
\]

\[
\frac{p_0 N(A_a = a)}{(1 - p_0)} \frac{P_{del}}{P_{add} q(A_a = a)}
\]

The factors in the first line of \( Q^* \) represent the ratio of the model likelihoods while the other two are respectively the ratio of the prior distributions and the ratio of the proposal distributions.

4.1 Unbiased Estimate to \( Q^* \) and \( 1/Q^* \)

Computing the partition functions in equation 8 is generally intractable. However, noticing that \( Z(Y_a = 1, A_a = a) \rightarrow Z(Y_a = 0) \), as \( a \rightarrow 0 \), the log-ratio of the two partition functions should be well approximated by a quadratic approximation at the origin when \( a \) is small. In this way
we reduce the problem of estimating the partition function to computing the first and second order derivatives of the log-partition function. We employ a second order Taylor expansion for the ratio of partition functions in $Q^\ast$ as follows:

$$
\frac{Z(Y_{\alpha} = 0)}{Z(Y_{\alpha} = 1, A_{\alpha} = a)} \overset{\text{def}}{=} R_{add}^{\text{na}} \approx \exp \left( -N a \frac{\partial \log(Z)}{\partial A_{\alpha}} |_{A_{\alpha}=0} - \frac{N a^2}{2} \frac{\partial^2 \log(Z)}{\partial A_{\alpha}^2} |_{A_{\alpha}=0} \right) \tag{9}
$$

We know that the $k$th order derivatives of the log-partition function of an MRF correspond to the $k$th order centralised moments (or cumulants) of the features, that is,

$$
\frac{\partial \log(Z)}{\partial A_{\alpha}} = E f_{\alpha}, \quad \frac{\partial^2 \log(Z)}{\partial A_{\alpha}^2} = \text{Var}(f_{\alpha}) \tag{10}
$$

Given a set of $n$ state samples $\tilde{x}^{(s)} \sim P(\mathbf{x} | Y_{\alpha} = 0)$ from the persistent Markov chains, we can compute an unbiased estimate of the mean and variance of $f_{\alpha}$ as the sample mean $\tilde{f}_{\alpha}$ and sample variance $S_{\alpha}^2 = \sum (f_{\alpha}(\tilde{x}^{(s)}) - \tilde{f}_{\alpha})^2 / (n-1)$ respectively. Consequently we obtain an estimate of $R_{add}$ by plugging $\tilde{f}_{\alpha}$ and $S_{\alpha}^2$ into equation (9), which is unbiased in the logarithmic domain, denoted as $\tilde{R}_{add}$.

An unbiased estimate in the logarithmic domain, unfortunately, is no longer unbiased once transformed to the linear domain. To correct the bias induced by the transformation, we take another Taylor expansion of $\tilde{R}_{add}$ around $a = 0$. After some derivation, we obtain an unbiased estimate of $\tilde{R}_{add}$ up to the second order of $a$ given by

$$
\tilde{R}_{add}(a) = \exp \left( -N a \tilde{f}_{\alpha} - \frac{N a^2}{2} S_{\alpha}^2 \right) \left( 1 + \frac{N a^2 S_{\alpha}^2}{2n} \right)^{-1}
$$

with variance:

$$
\text{Var}(\tilde{R}_{add} / R_{add}) = \frac{N^2 a^2}{n} \text{Var}(f_{\alpha}) + o(a^3) \tag{11}
$$

Similarly, we can also obtain an unbiased estimate, $\tilde{R}_{del}$, in $1/Q^\ast$ when considering deleting an edge, with the same formula as $\tilde{R}_{add}$ except that $a$ is replaced by $-a$ and the sample mean and variance are now estimated with respect to $P(\mathbf{x} | Y_{\alpha} = 1, A_{\alpha} = a)$. If we plug in $\tilde{R}_{add}$ (or $\tilde{R}_{del}$) into $Q_{add}$ (or $Q_{del}$) we get an unbiased estimate of the acceptance probability except when $Q^\ast$ (or $1/Q^\ast$) is close to 1 in which case the min operation causes extra bias. Since the variance can be computed as a function of $a$ in equation (11) we can estimate how large a jump range $\Delta$ can be used in order to keep the error in the acceptance probability negligible. A larger data set requires a smaller jump range or alternatively a larger sample set that grows quadratically with the size of the data set.

### 4.2 Optimal Proposal Distribution $q$

After plugging equations $9, 10$ and $5$ into equation $8$, we obtain the acceptance probability as a function of $a$:

$$
Q_{add} = \min \left\{ 1, \frac{h(a)}{q(A_{\alpha} = a)} \right\} \text{const}
$$

$$
h(a) = \exp \left( -\frac{a^2}{2} \left( \frac{1}{\sigma_0^2} + \text{Var}(f_{\alpha}) \right) + a N g_{\alpha} \right) \tag{12}
$$

where $g_{\alpha}$ and $\text{Var}(f_{\alpha})$ are defined at $\theta_{\alpha} = 0$. Clearly, the optimal proposal distribution in terms of minimal variance is given by the following truncated Gaussian distribution

$$
q_{opt}(A_{\alpha} = a | \theta) \propto h(a), \quad a \in [-\Delta_{\alpha}, \Delta_{\alpha}] \tag{13}
$$

When adding an edge, we have state samples $\tilde{x}^{(s)} \sim P(\mathbf{x} | Y_{\alpha} = 0)$. The expectation $E f_{\alpha}$ in $g_{\alpha}$ can be estimated by $\tilde{f}_{\alpha}(\{\tilde{x}^{(s)}\})$ and $\text{Var}(f_{\alpha})$ by $S_{\alpha}^2(\{\tilde{x}^{(s)}\})$. When deleting an edge, the samples are from $P(\mathbf{x} | Y_{\alpha} = 1, A_{\alpha} = a)$. We apply a quadratic approximation for $\log(Z(\theta_{\alpha}))$, use equation $10$ and estimate $\text{Var}(f_{\alpha}) \approx S_{\alpha}^2$ and $E f_{\alpha} \approx \tilde{f}_{\alpha} - a S_{\alpha}^2$.

### 4.3 Parallel Proposals

Since the most computationally demanding step is to obtain a set of state samples $\{\tilde{x}^{(s)}\}$, we want to reuse the samples whenever possible. Given that $\Delta_{\alpha}$ is small enough, the parameter value does not change much when we accept an “add or delete edge” move. We can thus assume that the distribution of $A_{\alpha}$ on an edge is not affected much by an accepted move on other edges. As a result we do not have to rerun the Gibbs sampler after every edge operation, and we can propose jumps for all $\{\alpha : |A_{\alpha}| < \Delta_{\alpha}\}$ in parallel, using the same set of samples. This reduces the computation time significantly.

### 5 Sampling for Hyper-parameters

Given $A$ and $Y$, the hyper-parameters are easy to sample when using conjugate priors:

$$
p_0 | Y \sim \text{Beta}(a + \sum_{\alpha} I(Y_{\alpha} = 1), b + \sum_{\alpha} I(Y_{\alpha} = 0))
$$

$$
\sigma_0^{-2} | Y, A \sim \Gamma(c + \frac{1}{2} \sum_{\alpha} I(Y_{\alpha} = 1), d + \frac{1}{2} \sum_{\alpha:Y_{\alpha}=1} A_{\alpha}^2) \tag{14}
$$

where $I$ is the indicator function.

The whole inference algorithm is summarized in Alg[1].

### 6 Experiments

#### 6.1 Datasets

We assess the performance of our proposed method on two simulated data sets and one real-world data set. For the
simulated data, we randomly generate sparse Ising models with binary ±1 states and with parameters sampled from a Gaussian distribution $N(0, \sigma^2)$ where $\sigma = 0.5$ for edges and 0.1 for node biases. These models are then converted to their equivalent Boltzmann machines with binary $\{0, 1\}$ states from which we draw exact samples. Two structures are considered: (1) a block model with 12 nodes equally divided into 3 groups. Edges are added randomly within a group with a probability of 0.8, and across groups with 0.1. Edges within a group are strong and positively coupled. There are 66 candidate edges with 20 edges in the ground truth model. (2) a $10 \times 10$ lattice with 4950 candidate edges and with 180 edges in the ground truth model.

For the real data, we use the MNIST digits image data. We convert the gray scale pixel values to binary values by thresholding at a value of 50, resize the images to a $14 \times 14$ scale, and then pick a $9 \times 12$ patch centered in each image where the average value of each pixel is in the range of $[10^{-4}, 1 - 10^{-4}]$. The last step is necessary because the other competing models do not have regularization on their biases, which will result in divergent parameter estimates for pixels that are always 0 or 1.

### 6.2 Model Specification

We compare our Bayesian structure learning algorithm with two other approaches. One is proposed in Wainwright et al. (2007) which recovers the neighbourhood of nodes by training separate $L_1$-regularized logistic regressions on each variable. While its goal is edge detection, we can also use it as a parameter estimation method with two output models, “Wain Max” and “Wain Min”, as defined in Höfling and Tibshirani (2009). We implement the “Wain Max/Min” methods with the Lasso regularized generalized linear model package of Friedman et al. (2010).6

6Code provided at stat.stanford.edu/~tibs/glmnet-matlab

The other approach is one of the several variants of $L_1$-regularized MLE methods which use a pseudo-likelihood approximation (Höfling and Tibshirani, 2009), denoted as “MLE”.[3]

For our Bayesian model, we consider two schemes to specify a model for prediction. One is the fully Bayesian approach, referred as “Bayes”, in which we random pick 100 model samples in the Markov chain and approximate the Bayesian model by a mixture model of these 100 components. The other one is to obtain a single model by applying a threshold of 0.5 to $P(Y_{i,j} | D)$ and estimate the posterior mean of the included edges, referred to as “Bayes PM”.

The performance of the Bayesian model is insensitive to the choice of hyper-parameters. We simply set $a = b = c = d = 5$ for $p_0$ and $\sigma_0$, and $\sigma_0 = 10$ across all experiments. For the parameters of the MCMC method, we also use a common setting. We use a diagonal approximation to the feature covariance $\text{Cov}_f$ and thereby the preconditioning matrix $C$. We set the sample size $n = 100$, number of Gibbs sampling steps $N_{\text{Gibbs}} = 1$, LMC step size $\varepsilon = 10^{-3}$, and momentum refreshment rate $\alpha = 0.9$. The RJMCMC proposal width is set as $\Delta = 0.01/\sqrt{N\text{Var}_f}$ to achieve a small estimation error as in equation [11]. For each experiment, we run the MCMC algorithm to collect $10K$ samples with a subsampling interval of 1000. Since the exact partition function can be computed on the small block model, we also run an exact MCMC, “Bayes Exact”, with an exact gradient and accept/reject decision as well as larger values in $\varepsilon$, $\alpha$, and $\Delta$ than the approximate MCMC.

Different levels of sparsity have to be considered in $L_1$-based methods for an optimal regularization strength. For the Bayesian method, we learn a single sparsity level. However, for the sake of comparison, we also consider a method with $p_0$ as a parameter and vary it between (0, 1) to induce different sparsity, referred to with a suffix “$p_0$”.

### 6.3 Accuracy of Inference

We first evaluate the validity of the proposed MCMC method on the block data where exact inference can be carried out. The marginal posterior distribution of an edge parameter, $\theta_{i,j}$, is a mixture of a point mass at zero and a continuous component with a single mode. Figure 5 shows the histogram of samples in the continuous component of four randomly picked $\theta_{i,j}$’s. The title above each plot is the posterior probability of the edge $(i,j)$ being present in the model, i.e. $\theta_{i,j} \neq 0$ or $Y_{i,j} = 1$. In this figure, the marginal distribution estimated from the approximate MCMC method matches the distribution from “Bayes Exact” very well. For a more comprehensive comparison, we run “Bayes $p_0$” and “Bayes Exact $p_0$” methods, and vary the value of $p_0$ from $10^{-4}$ to 0.99 to achieve different levels of sparsity. We estimate and collect across different $p_0$
models, and compute log the block model and a choose a group of variables, which is all the variables in
For each data case in the validation set, we randomly
the conditional marginal log-likelihood in Lee et al. (2006).
Since computing the log-likelihood of a MRF is in general
intractable, we use the conditional log-likelihood (CLL) on
edges. The quality of the estimated models is evaluated
simulated data sets for two tasks: structure recovery and

6.4 Simulated Data
We then compare the performance of various methods on
simulated data sets for two tasks: structure recovery and
model estimation. The accuracy of recovering the true
structure is measured by the precision and recall of the true
edges. The quality of the estimated models is evaluated
by the predictive performance on a held-out validation set.
Since computing the log-likelihood of a MRF is in general
intractable, we use the conditional log-likelihood (CLL) on
a group of variables instead, which is a generalization of
the conditional marginal log-likelihood in Lee et al. (2006).
For each data case in the validation set, we randomly
choose a group of variables, which is all the variables in
the block model and a 3 × 3 grid in the lattice and MNIST
models, and compute log P(\{x_i\}_{i \in \text{group}}|\{x_j\}_{j \not\in \text{group}}).

We train models on the simulated data ranging between 50
and 1000 items. For the Bayesian models, we remove an
edge if the posterior probability \( P(Y_{i,j} = 1|\mathcal{D}) < 0.5.\)

values the posterior probability of \( \theta_{i,j} \neq 0, \) posterior mean
and standard deviation of \( \theta_{i,j} \) when \( Y_{i,j} = 1. \) The training data is
from the block model with \( N = 100. \)

Figure 4: Comparison of “Bayes \( p_0 \)” and “Bayes Exact \( p_0 \)” on
posterior probability of \( Y_{i,j} = 1, \) and the posterior mean and
standard deviation of \( \theta_{i,j} \) when \( Y_{i,j} = 1. \) The training data is
from the block model with \( N = 100. \)

Figure 5 shows typical precision-recall curves for different
methods. It turns out that all the models with a sparsity
tuning parameter perform similarly across all the training
sets. The “Bayes” model tends to find a structure with high
precision.

We then consider the average CLL as a function of the edge
density of a model. The edge density is defined as the per-
centage of edges present in a model, i.e., \( 1 – \text{sparserity}. \) For
the fully Bayesian model, we measure the average density
in the Markov chain. In fact, the variance of the edge den-
sity is usually small, suggesting that most model samples
have about the same number of edges. We show the av-
erage CLL of different models trained with 100 data cases
in Figure 6. The results on other data sizes are omitted
because they have the same tendency. All the Bayesian
models give robust prediction performance in the sparse
and dense model ranges and “Bayes \( p_0 \)” outperforms all
the other methods with a tunable sparsity parameter for al-
most all settings of \( p_0. \) Moreover, the curve of “Bayes \( p_0 \)”
peaks at the true density level. In contrast, \( L_1 \) methods
would underfit to the data for sparse models or overfit for
dense models. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-
els. “MLE” performs better than “Wain” mod-

Figure 3: Histogram of the parameter samples of four randomly
picked edges from “Bayes” and “Bayes Exact” when \( Y_{i,j} = 1. \)
The training data is from the block model with \( N = 100. \)

Figure 5: Precision-Recall Curves for the lattice data with 100
data cases.
The Bayesian models, “Bayes” and “MLE” are similar to “Wain Max”. 65% (bottom, ~) models (top, edge density true model. The parameters in “Wain Max” are too small in sparse model and sorted along the x-axis by the absolute value in the parameters are plotted only if they are not zero in at least one model, which consequently brings down the large value, it forces a lot of non-existing edges to be included in the model, which brings down the posterior distribution of σ₀. This results in too small values on real edges as shown in the inset of Figure 8 and thereby a decrease in the model predictive accuracy.

However, this “misbehavior” in return just suggests the ability of our Bayesian model to learn the true structure. Once we release ρ₀ through a hierarchical prior, the model will abandon these improper values in ρ₀ and automatically find a good structure and parameters. The vertical line in Figure 6 indicates the sparsity of the true model. Both “Bayes” and “Bayes PM” find sparsity levels very close to the true value, while L₁-based methods are under-fitted at that same level as shown in the upper panel of Figure 8. We show the joint performance of edge detection and parameter learning in Figure 9 where the performance of edge detection is summarized by the F1 score (the harmonic mean of the precision and recall). The Bayesian models with a hierarchical ρ₀ prior achieve both a high F-1 score and CLL value near the upper right corner.

6.5 MNIST Data

Since there does not exist ground truth in the model structure of the MIST data set, we evaluate how well we can learn a sparse model for prediction. Figure 10 shows the average CLL on 10K test images with a model trained on
Figure 10: Mean and standard deviation of average CLL versus edge density on MNIST with 100 and 1000 data cases.

100 and 1000 images respectively. In the sparse and dense model ranges, we observe again a better performance of “Bayes” than $L_1$-based methods. “Bayes PM” also shows robustness to under/over-fitting although it seems that simply computing the posterior mean does not provide sufficiently good model parameters in the median density range.

To get a more intuitive comparison about the quality of learned sparse models, we train models on 1000 images by different methods with a density of 0.2 and then run Gibbs sampling to draw 36 samples from each model. The images are shown in Figure 11. While it is hard to get good reconstruction using a model without hidden variables, the Bayesian methods produce qualitatively better images than competing methods, even though “Bayes PM” does not have higher CLL than “MLE” at this level.

A common limitation of learning Bayesian models with the MCMC inference method is that it is much slower than training a model with a point estimate. However, as shown in the experiments, the Bayesian methods are able to learn a good combination of parameters and a structure without the need to tune hyper-parameters through cross validation. Also, the Bayesian methods learn sparser models than $L_1$-based methods without sacrificing predictive performance significantly. Because the computational complexity of inference grows exponentially with the maximum clique size of MRFs, $L_1$-based models at their optimal (not so sparse) regularization level can in fact become significantly more computationally expensive than their Bayesian counterparts at prediction time. Turning up the regularization will result in sparser models but at the cost of under-fitting the data and thus sacrificing predictive accuracy.

7 Discussion

We propose Bayesian structure learning for MRFs with a spike and slab prior. An approximate MCMC method is proposed to achieve effective inference based on Langevin dynamics and reversible jump MCMC. As far as we known this is the first attempt to learn MRF structures in the fully Bayesian approach using spike and slab priors. Related work was presented in Parise and Welling (2006) with a variational method for Bayesian MRF model selection. However this method can only compare a given list of candidate models instead of searching in the exponentially large structure space.

The proposed MCMC method is shown to provide accurate posterior distributions at small step sizes. The selective shrinkage property of the spike and slab prior enables us to learn an MRF at different sparsity levels without noticeably suffering from under-fitting or over-fitting even for a small data set. Experiments with simulated data and real-world data show that the Bayesian method can learn both an accurate structure and a set of parameter values with strong predictive performance. In contrast the $L_1$-based methods could fail to accomplish both tasks with a single choice of the regularization strength. Also, the performance of our Bayesian model is largely insensitive to the choice of hyper-parameters. It provides an automated way to choose a proper sparsity level, while $L_1$ methods usually rely on cross-validation to find their optimal regularization setting.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. 0914783, 0928427, 1018433.
References


