Model Averaging with Discrete Bayesian Network Classifiers

Denver Dash
Decision Systems Laboratory *
Intelligent Systems Program
University of Pittsburgh
Pittsburgh, PA 15213

Gregory F. Cooper
Center for Biomedical Informatics
University of Pittsburgh
Pittsburgh, PA 15213

Abstract

This paper considers the problem of performing classification by model-averaging over a class of discrete Bayesian network structures consistent with a partial ordering and with bounded in-degree k. We show that for N nodes this class contains in the worst-case at least $\Omega(\binom{N/2}{k}^{N/2})$ distinct network structures, but we show that this summation can be performed in $O(\binom{N}{k} \cdot N)$ time. We use this fact to show that it is possible to efficiently construct a single directed acyclic graph (DAG) whose predictions approximate those of exact model-averaging over this class, allowing approximate model-averaged predictions to be performed in $O(N)$ time. We evaluate the procedure in a supervised classification context, and show empirically that this technique can be beneficial for classification even when the generating distribution is not a member of the class being averaged over, and we characterize the performance over several parameters on simulated and real-world data.

1 Introduction

The general supervised classification problem seeks to create a model based on labelled data D, which can be used to classify future vectors of features $\boldsymbol{F} = \{F_1, F_2, \ldots, F_N\}$ into one of various classes of interest. A Bayesian network (BN) classifier is a probabilistic model that accomplishes this goal by explicating causal interactions/conditional independencies between features in $\boldsymbol{F}$. The simplest Bayesian network classifier for this task is the naive classifier, which, without inferring any structural information from the database, can still perform surprisingly well at the classification task [Domingos and Pazzani, 1997]. More sophisticated algorithms for learning BNs from data [Verma and Pearl, 1991, Cooper and Herskovits, 1992, Spirtes *et al.*, 1993, Heckerman *et al.*, 1995, Friedman *et al.*, 1997] can also be used effectively to construct a BN model by extracting information about conditional independencies from D to build more accurate structural models. Classification using a single Bayesian network model for a fixed number of classes when the feature vector is completely instantiated can be performed in $O(N)$ time.

This procedure of selecting a single model for classification has the potential drawback of over-fitting the data however, leading to poor generalization capability. A strictly Bayesian approach, averaging classifications over all models weighted by their posterior probability given the data, has been shown to reduce overfitting and provide better generalization [Madigan and Raftery, 1994]. Unfortunately, the space of network structures is super-exponential in the number of model variables, and thus an exact method for full model-averaging is likely to be intractable.

In this paper we consider the possibility of performing exact and approximate model-averaging (MA) over a particular class of structures rather than over the general space of DAGs. Recently Dash and Cooper [2002], demonstrated that exact model averaging over the restricted class of naive networks could be performed by a simple re-parametrization of a naive network, and they showed that this technique consistently outperformed a single naive classifier with the standard parametrization. The present paper generalizes that result; in particular, we show that exact model averaging over the class of BN structures consistent with a partial ordering π and with bounded in-degree k, despite its super-exponential size, can be performed with relatively small time and space restrictions.

Methods for approximate MA classification using both selective pruning [Madigan and Raftery, 1994, Volinsky, 1997] and Monte-Carlo [Madigan and York, 1995]

*This work was partially performed during a summer internship at the Machine Learning and Perception group, Microsoft Research, Cambridge, UK.

techniques exist and have shown to improve prediction tasks; however these methods do not possess the extent of time efficiency of inference that we seek.

Friedman and Koller [2000] studied the ability to estimate structural features of a network (for example the probability of an arc from X_i to X_j) by performing a MCMC search over orderings of nodes. Their method relied on a decomposition, which they credited to Buntine [1991], that we extend in order to prove our key theoretical result. We discuss this issue in detail in Section 3.3. Their work differs from ours in two key respects: (1) Their approach does not capture the single-network (and thus the efficiency of calculation) approximation to the MA problem, and (2) They perform model averaging only to calculate the probabilities of structural features, explicitly not for classification.

Meila and Jaakkola [2000] discuss the ability to perform exact model averaging over all trees. They also use similar assumptions and similar decompositions that we use; however our calculation is more general in allowing nodes to have more than one parent, but it is less general in that it assumes a partial-ordering of the nodes. Their approach also does not allow for $O(N)$ model-averaged classifications.

Our primary contributions in this paper are as follows: (1) we extend the factorization of conditionals to apply to the task of classification, (2) we show that MA calculations over this class can be approximated by a single network structure S^* which can be constructed efficiently, thereafter allowing approximate MA predictions to be performed in $O(N)$ time, and (3) we demonstrate empirically that, especially when the number of records is small compared to the size of the network, using this technique for classification can be beneficial compared to (a) a single naive classifier, (b) single networks learned from data using a greedy Bayesian learning algorithm and (c) exact model averaging over naive classifiers.

In Section 2 we formally frame the problem and state our assumptions and notation. In Section 3 we derive the MA solution and show that the MA predictions are approximated by those of a single structure bearing a particular set of parameters. In Section 4 we present the experimental comparisons, and in Section 5 we discuss our conclusions and future directions.

2 Assumptions and Notation

The general supervised classification problem can be framed as follows: Given a set of features $\boldsymbol{F} = \{F_1, F_2, \ldots, F_N\}$, a set of classes $\mathcal{C} = \{C_1, C_2, \ldots C_{N_c}\}$, and a labelled database $D = \{D_1, D_2, \ldots, D_R\}$, construct a model to predict into which class future feature vectors are most likely to reside.

We use the notation X_i to refer to the nodes when we need to have a uniform notation; using the convention that, $X_i \equiv F_i$ and $X_0 \equiv C$, and we use $\boldsymbol{X}$ to denote the collective set of nodes in the network. A directed graph $G(\boldsymbol{X})$ is defined as a pair $\langle \boldsymbol{X}, \boldsymbol{E} \rangle$, where $\boldsymbol{E}$ is a set of directed edges $X_i \rightarrow X_j$, such that $X_i, X_j \in \boldsymbol{X}$.

We assume that each node X_i is a discrete multinomial variable with r_i possible states $\{x_i^1, x_i^2, \ldots, x_i^{r_i}\}$. We use $\boldsymbol{P}_i$ to denote the parent set of X_i, and we let q_i denote the number of possible joint configurations of parents for node X_i, which we enumerate as $\{p_i^1, p_i^2, \ldots, p_i^{q_i}\}$. We use θ_{ijk} to denote a single parameter of the network: $\theta_{ijk} = P(X_i = x_i^k \mid \boldsymbol{P}_i = \boldsymbol{p}_i^j)$, and the symbol $\boldsymbol{\theta}$ to denote the collective parameters of the network. In general we use the common (ijk) coordinates notation to identify the *k-th* state and the *j-th* parent configuration of the *i-th* node in the network. We take the common assumptions in BN structure learning of Dirichlet priors, parameter independence, and parameter modularity [Heckerman *et al.*, 1995]. We also assume that database D consists of complete labelled instances.

3 Theoretical Results

In this section we show how to efficiently calculate the quantity $P(\boldsymbol{X} = \boldsymbol{x} \mid D)$ averaged over the class of structures we are considering.

3.1 Fixed Network Structure

For a fixed network structure S and a fixed set of network parameters $\boldsymbol{\theta}$, the quantity $P(\boldsymbol{X} = \boldsymbol{x} \mid S, \boldsymbol{\theta})$ can be calculated in $O(N)$ time:

$$P(\boldsymbol{X} = \boldsymbol{x} \mid S, \boldsymbol{\theta}) = \prod_{i=0}^{N} \theta_{iJK}, \tag{1}$$

where all (j, k) coordinates are fixed by the configuration of $\boldsymbol{X}$ to the value $(j, k) = (J, K)$.

When, rather than a fixed set of parameters, a database D is given, from an ideal Bayesian perspective it is necessary to average over all possible configurations of the parameters $\boldsymbol{\theta}$:

$$\begin{aligned} P(\boldsymbol{X} = \boldsymbol{x} \mid S, D) &= \int P(\boldsymbol{X} = \boldsymbol{x} \mid S, \boldsymbol{\theta}) \cdot P(\boldsymbol{\theta} \mid S, D) \cdot d\boldsymbol{\theta} \\ &= \int \prod_{i=0}^{N} \theta_{iJK} \cdot P(\boldsymbol{\theta} \mid S, D) \cdot d\boldsymbol{\theta} \end{aligned}$$

where the second line follows from Equation 1. Given the assumption of parameter independence and Dirichlet priors, this quantity can be written just in terms

of sufficient statistics and Dirichlet hyperparameters [Cooper and Herskovits, 1992, Heckerman *et al.*, 1995]:

$$P(\boldsymbol{X} = \boldsymbol{x} \mid S, D) \quad = \quad \prod_{i=0}^{N} \frac{\alpha_{iJK} + N_{iJK}}{\alpha_{iJ} + N_{iJ}}, \qquad (2)$$

where we have used the notation that $Q_{ij} = \sum_k Q_{ijk}$[1]. Comparing this result to Equation 1 illustrates the well-known result that a single network with a fixed set of parameters $\hat{\boldsymbol{\theta}}$ given by

$$\hat{\theta}_{ijk} = \frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}} \qquad (3)$$

will produce predictions equivalent to those obtained by averaging over all parameter configurations. We refer to $\hat{\theta}_{ijk}$ as the *standard parametrization*.

3.2 Averaging Structural Features with a Fixed Ordering

The decomposition by Buntine used by Friedman and Koller was a dynamic programming solution which calculated, with relative efficiency, the posterior probability of a structural feature (for example a particular arc $X_L \rightarrow X_M$) averaged over all in-degree-bounded networks consistent with a fixed ordering. Here we re-derive the result.

The derivation required an additional assumption, labelled "structure modularity" by Friedman and Koller:

Assumption 1 (Structure modularity) *The prior of a structure S, $P(S)$, can be factorized according to the network:*

$$P(S) \propto \prod_{i=0}^{N} p_s(X_i, \boldsymbol{P}_i), \qquad (4)$$

where $p_s(X_i, \boldsymbol{P}_i)$ is some function that depends only on the local structure (X_i and $\boldsymbol{P}_i$).

Obviously the uniform distribution which is $O(1)$ will satisfy this assumption, as will other common network prior forms.

The posterior probability $P(X_L \rightarrow X_M \mid D)$ can be written as:

$$P(X_L \rightarrow X_M \mid D) = \kappa \sum_S \delta(X_L \rightarrow X_M \in S) \cdot P(D \mid S) \cdot P(S) \qquad (5)$$

where $\kappa = 1/P(D)$ is a constant that depends only on the database, and $\delta(X)$ is the Kronecker delta function:

$$\delta(X) = \begin{cases} 1 & \text{if } X = \mathit{true} \\ 0 & \text{otherwise} \end{cases}$$

[1]More generally, we will use $Q_{ij} = \sum_k Q_{ijk}$ for a quantity Q.

Given the assumptions of complete data, multinomial variables, Dirichlet priors and parameter independence, the marginal likelihood $P(D \mid S)$ can be written just in terms of hyperparameters and sufficient statistics [Cooper and Herskovits, 1992, Heckerman *et al.*, 1995]:

$$P(D \mid S) = \prod_{i=0}^{N} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}. \qquad (6)$$

Given Assumption 1 and Equation 6, Equation 5 can be written as:

$$P(X_L \rightarrow X_M \mid D) = \kappa \sum_S \prod_{i=0}^{N} \rho_{iLM} \qquad (7)$$

where the ρ_{iLM} functions are given by:

$$\rho_{iLM} = \delta[M \neq i \vee X_L \in \boldsymbol{P}_i] \cdot p_s(X_i, \boldsymbol{P}_i) \cdot \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}, \qquad (8)$$

and can be calculated using information that depends only on nodes X_i and $\boldsymbol{P}_i$.

Buntine considered a total ordering on the nodes, but generalization to a partial ordering is straightforward. For a given partial ordering π and a particular node X_i we want to enumerate all of X_i's possible parent sets up to a maximum size k. To this end, we will typically use the superscript ν to index the different parent sets. For example, four nodes partitioned as $\langle\{X_1, X_3\}, \{X_2, X_4\}\rangle$ and a maximum in-degree $k = 2$ might yield the following enumeration of parent sets for X_2: $\{\boldsymbol{P}_2^0 = \emptyset, \boldsymbol{P}_2^1 = \{X_1\}, \boldsymbol{P}_2^2 = \{X_3\}, \boldsymbol{P}_2^3 = \{X_1, X_3\}\}$, any of which we might refer to as $\boldsymbol{P}_2^\nu$. We assume the hyperparameters α_{ijk}^ν for any parent set $\boldsymbol{P}_i^\nu$ can be calculated in constant time. For example the K2 scoring metric [Cooper and Herskovits, 1992], which sets $\alpha_{ijk} = 1$ for all (ijk), will satisfy this requirement. The class of models consistent with π with bounded in-degree of k we denote as $\mathcal{L}_k(\pi)$:

Definition 1 (Multi-level k-graphs, $\mathcal{L}_k(\pi)$) *For a given integer $k \leq N$ and a given partial ordering π of $\boldsymbol{X}$, a DAG $G = \langle \boldsymbol{X}, \boldsymbol{E} \rangle$ is a multi-level k-graph with respect to π (denoted as $\mathcal{L}_k(\pi)$) if arcs are directed down levels and no variable has more than k parents: $X_i \rightarrow X_j \in \boldsymbol{E} \Rightarrow \mathit{Level}(X_i) < \mathit{Level}(X_j)$, and $X_i \in \boldsymbol{X} \Rightarrow |\boldsymbol{P}_i| \leq k$.*

When model-averaging over the class $\mathcal{L}_k(\pi)$, each node at level l can choose from all the nodes in layers $l' < l$ at most k parents. For $k < N/2$, Equation 7 in the worst-case includes a summation over $\Omega\left[\binom{N/2}{k}^{N/2}\right]$ network

structures (this worst-case corresponding to two layers, each with $N/2$ nodes). However, the following theorem shows that Equation 7 can be calculated with relative efficiency:

Theorem 1 *For N variables with a maximum number of states per variable given by N_f and a database of N_r records, the right-hand side of Equation 7 can be calculated in $O(\binom{N}{k} \cdot N \cdot N_r \cdot {N_f}^k)$ time and using $O(\binom{N}{k} \cdot N \cdot {N_f}^k)$ space if the summation is restricted to network structures in $\mathcal{L}_k(\pi)$.*

Proof: We use μ_i to denote the maximum number of parent sets available to node X_i. Expanding the sum in Equation 7 and using the notation ρ^{ν}_{iLM} to denote ρ_{iLM} for the νth parent set $\boldsymbol{P}_i^{\nu}$, yields:

$$\left.\begin{array}{rl} & P(X_L \rightarrow X_M \mid D) \propto \\ & \quad \rho^0_{0LM} \cdot \rho^0_{1LM} \cdot \ldots \cdot \rho^0_{NLM} \\ + & \quad \rho^1_{0LM} \cdot \rho^0_{1LM} \cdot \ldots \cdot \rho^0_{NLM} \\ \vdots & \qquad \vdots \\ + & \quad \rho^{\mu_0}_{0LM} \cdot \rho^0_{1LM} \cdot \ldots \cdot \rho^0_{NLM} \\ + & \quad \rho^0_{0LM} \cdot \rho^1_{1LM} \cdot \ldots \cdot \rho^0_{NLM} \\ \vdots & \qquad \vdots \\ + & \quad \rho^{\mu_0}_{0LM} \cdot \rho^{\mu_1}_{1LM} \cdot \ldots \cdot \rho^{\mu_N}_{NLM} \end{array}\right\} \Omega\left[\binom{N/2}{k}^{N/2}\right] \text{ terms.}$$

We define the symbol Σ^{LM}_m to denote the structure sum of the product up to and including the m-th node:

$$\begin{array}{rcl} \Sigma^{LM}_m & \equiv & \rho^0_{0LM} \cdot \rho^0_{1LM} \cdot \ldots \cdot \rho^0_{mLM} \\ & + & \rho^1_{0LM} \cdot \rho^0_{1LM} \cdot \ldots \cdot \rho^0_{mLM} \\ & \vdots & \qquad \vdots \\ & + & \rho^{\mu_0}_{0LM} \cdot \rho^{\mu_1}_{1LM} \cdot \ldots \cdot \rho^{\mu_m}_{mLM} \end{array}$$

Using this notation, the following recursion relation can be derived:

$$\Sigma^{LM}_i = \Sigma^{LM}_{i-1} \cdot \sum_{\nu=1}^{\mu_i} \rho^{\nu}_{iLM}, \qquad \Sigma^{LM}_{-1} = 1$$

Finally, expanding out the recurrence relation yields the expression for $P(X_L \rightarrow X_M \mid D)$:

$$P(X_L \rightarrow X_M \mid D) = \kappa \prod_{i=0}^{N} \sum_{\nu=1}^{\mu_i} \rho^{\nu}_{iLM} \tag{9}$$

Once the ρ^{ν}_{iLM} terms are calculated, the right-hand-side of Equation 9 can be performed in $O(N \cdot \binom{N}{k})$ time. Calculating the ρ terms themselves requires the calculation of one hyperparameter α^{ν}_{ijk} and sufficient statistic N^{ν}_{ijk} for each parameter of the network, each of which can be calculated in $O(N_r)$ time. To calculate the complete set for all (j,k) combinations thus requires $O(\binom{N}{k} \cdot N \cdot N_r \cdot {N_f}^k)$ time and $O(\binom{N}{k} \cdot N \cdot {N_f}^k)$ space (because there are $O(N_f^k)$ network parameters per node). □

In fact, the ${N_f}^k$ characterization is a loose upper-bound, and can likely be reduced by compressing contingency tables.

3.3 Model Averaging for Prediction

Here we show that a similar dynamic programming solution exists for model-averaging the quantity $P(\boldsymbol{X} = \boldsymbol{x} \mid D)$ over the class $\mathcal{L}_k(\pi)$.

This quantity can be written as:

$$P(\boldsymbol{X} = \boldsymbol{x} \mid D) = \kappa \sum_{S} \prod_{i=0}^{N} \hat{\theta}_{iJK} \cdot P(D \mid S) \cdot P(S), \tag{10}$$

where $\hat{\theta}_{iJK}$ are the standard parameters given in Equation 3. Given structure modularity and Equation 6, Equation 10 can be written in a form very similar to Equation 7:

$$P(\boldsymbol{X} = \boldsymbol{x} \mid D) = \kappa \sum_{S} \prod_{i=0}^{N} \hat{\rho}_{iJ^S_{\boldsymbol{x}}K_{\boldsymbol{x}}} \tag{11}$$

where here the $\hat{\rho}_{iJ^S_{\boldsymbol{x}}K_{\boldsymbol{x}}}$ functions are given by:

$$\begin{aligned} \hat{\rho}_{iJ^S_{\boldsymbol{x}}K_{\boldsymbol{x}}} = \hat{\theta}_{iJ^S_{\boldsymbol{x}}K_{\boldsymbol{x}}} \cdot p_s(X_i, \boldsymbol{P}_i) \cdot \\ \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}. \end{aligned} \tag{12}$$

We have taken the trouble to subscript the indices J and K from Equation 1. Both are indexed with an $\boldsymbol{x}$ to indicate that they are fixed by the particular configuration of $\boldsymbol{X}$, and J is indexed by S to emphasize that the value of the parent configuration index depends on the structure of the network. Although this notation may seem cumbersome, we hope it clarifies the analysis later.

The $\hat{\rho}_{iJ^S_{\boldsymbol{x}}K_{\boldsymbol{x}}}$ functions again can be calculated using only information local to node X_i and $\boldsymbol{P}_i$. Following a derivation identical to that for averaging $P(X_L \rightarrow X_M \mid D)$ in Section 3.2, yields the result:

$$P(\boldsymbol{X} = \boldsymbol{x} \mid D) = \kappa \prod_{i=0}^{N} \sum_{\nu=1}^{\mu_i} \hat{\rho}^{\nu}_{iJ^{\nu}_{\boldsymbol{x}}K_{\boldsymbol{x}}}. \tag{13}$$

Here the S index has been replaced with a ν indicating which parent set for node X_i is being considered. Once again this summation can be performed in $O(\binom{N}{k} \cdot N \cdot N_r \cdot {N_f}^k)$ time and $O(\binom{N}{k} \cdot N \cdot {N_f}^k)$ space.

3.4 Approximate Model Averaging

The derivation in the preceding section is an extension of the concept underlying Buntine's dynamic programming solution. The functional form of the above solution allows us to easily prove the following theorem:

Theorem 2 *There exists a single Bayesian network model $M^* = \langle S^*, \boldsymbol{\theta}^* \rangle$ that will produce predictions equivalent to those produced by model averaging over all models in $\mathcal{L}_k(\pi)$.*

Proof: Let S^* be defined so that each node X_i has the parent set $\boldsymbol{P}_i^* = \bigcup_{\nu=1}^{\mu_i} \boldsymbol{P}_i^\nu$, and let $\boldsymbol{\theta}^*$ be defined by:

$$\theta_{ijk}^* = \frac{1}{\kappa\sqrt{N}} \sum_{\nu=1}^{\mu_i} \hat{\rho}_{iJ_j^\nu k}^\nu \tag{14}$$

where the $\boldsymbol{x}$ subscript for $J_{\mathbf{x}}^\nu$ has now been replaced with a j and $K_{\mathbf{x}}$ has been replaced with a k subscript, because we are now considering a particular coordinate (ijk). It can be seen by direct comparison that the single network prediction using M^* and Equation 1 will yield Equation 13. □

If we define functions $f(X_i, \boldsymbol{P}_i^\nu \mid D)$ such that $f(X_i, \boldsymbol{P}_i^\nu \mid D) = \hat{\rho}_{iJ_j^\nu k}^\nu / \hat{\theta}_{iJ_j^\nu k}^\nu$, then Equation 14 can be written as:

$$\theta_{ijk}^* = \frac{1}{\kappa\sqrt{N}} \sum_{\nu=1}^{\mu_i} \hat{\theta}_{iJ_j^\nu k}^\nu \cdot f(X_i, \boldsymbol{P}_i^\nu \mid D) \tag{15}$$

The functions $f(X_i, \boldsymbol{P}_i^\nu \mid D)$ do not depend on the indices J_j^ν and k, and they are proportional to the local contribution of the posterior probability that the parent set of X_i is in fact $\boldsymbol{P}_i^\nu$. Equation 15 thus provides the interpretation that M^* represents a structure-based smoothing where each standard parameter $\hat{\theta}_{ijk}^\nu$ is weighted based on the likelihood that $\boldsymbol{P}_i^\nu$ is the true parent set of X_i.

Theorem 2 implies that, rather than performing the $O(\binom{N}{k} \cdot N)$ summation in Equation 13 for each case to be classified, in principle we need only construct a single model M^* and use standard $O(N)$ Bayesian network inference for each case.

A serious practical difficulty is that this approach requires in the worst case the construction of a completely-connected Bayesian network and is thus intractable for reasonable size N. An obvious pruning strategy, however, is to truncate the sum in Equation 14 to include no more than n parents. If we reorder the possible parent sets for node X_i as $O_P \equiv \{\boldsymbol{P}_i^1, \ldots, \boldsymbol{P}_i^{\mu_i}\}$ such that $f(X_i, \boldsymbol{P}_i^\nu \mid D) > f(X_i, \boldsymbol{P}_i^\lambda \mid D)$ only if $\nu < \lambda$, then a reasonable approximation for $\boldsymbol{P}_i^*$ can be constructed by the following procedure:

Procedure 1 (Approximate P_i^* construction)
Given: n and O_P.

1. *Let* $\boldsymbol{P}_i^* = \emptyset$
2. *For* $\nu = 1$ *to* μ_i,
   if $|\boldsymbol{P}_i^* \cup \boldsymbol{P}_i^\nu| \leq n$, *let* $\boldsymbol{P}_i^* = \boldsymbol{P}_i^* \cup \boldsymbol{P}_i^\nu$,
   else continue.

We denote the class of structures being averaged over using this procedure as $\mathcal{L}_k^n(\pi \mid D)$, and we call the method *Approximate Model Averaging* (AMA). Obviously $\lim_{n \to N} \mathcal{L}_k^n(\pi \mid D) = \mathcal{L}_k(\pi)$. Furthermore, we empirically show in Section 4 that the loss in ROC area, ϵ, due to this approximation for $n \geq 10$ lies around $-0.6\% \leq \epsilon \leq 0.6\%$ with 99% confidence for $N \leq 100$ and for a wide range of other parameters.

4 Experimental Tests

In this section we describe several experimental investigations that were designed to test the performance of (AMA) on distributions that do not necessarily fall into $\mathcal{L}_k(\pi)$. We first generate synthetic data to allow us to more extensively vary parameters and to generate statistically significant results, then we perform tests on several real-world machine learning data sets.

4.1 Experimental Setup

There are at least five parameters for which we sought to characterize the performance of AMA predictions: the number of nodes N, the approximation limit n on number of nodes, the maximum in-degree ("density") K of the generating network, the maximum number of parents k allowed in $\mathcal{L}_k(\pi)$, and the number of records N_r. It is beyond the scope of this paper to present a comprehensive comparison over this full five-dimensional space; however, here we sample their settings to provide insight into the dependence of the results on these parameters.

We compared AMA to three other Bayesian network classifiers: a single naive network (SNN) with the standard parametrization [Domingos and Pazzani, 1997], a model that model-averaged over all naive structures (NMA) [Dash and Cooper, 2002], and a non-restricted two-stage thick-thin greedy search (GTT) over the space of DAGs. GTT starts with an empty graph and repeatedly adds the arc (without creating a cycle) that maximally increases the marginal likelihood $P(D \mid S)$ until no arc addition will result in a positive increase, then it repeatedly removes arcs until no arc deletion will result in a positive increase in $P(D \mid S)$. GTT assumes no ordering on the nodes.

All abbreviations and symbols are summarized in Table 1 as a reference for the reader.

The inner-loop of each test performed basically the same procedure: Given the five parameters $\{N, N_r, K, n, k\}$ and a total number of trials N_{trials}, we did the following:

Procedure 2 (Basic testing loop)
Given: N, N_r, N_{test}, K, n, k, and N_{trials}.

Symbol	*Description*
N	Number of nodes
N_r	Number of training records
K	Maximum in-degree of generating graphs
k	Maximum in-degree in $\mathcal{L}_k$
n	Maximum in-degree in summary MA network.
SNN	Single naive network
NMA	Naive model averaging
GTT	Greedy thick-thin
AMA	Approximate model averaging

Table 1: Table of symbols.

Do:

1. *Generate N_{trials} random graphs $G(N, K)$.*
2. *For each graph $G(N, K)$ do:*
   (a) *Generate N_r training records and N_{test} test records.*
   (b) *Train two classifiers to be compared M_1 (typically the AMA classifier) and M_2 (the classifier to be compared) on the training records.*
   (c) *Test M_1 and M_2 on the test data, measuring the ROC areas R_1 and R_2, respectively, of each.*
   (d) *Calculate the quantity $\delta = \frac{R_1 - R_2}{T - R_2}$, where T is the ROC area of a perfect classifier (i.e., 1 if the axes are normalized).*
3. *Average δ over all N_{trials}.*

The performance metric δ indicates what percentage of M_2's missing ROC area is captured by M_1.

For some experiments it was necessary to generate networks randomly from a uniform distribution over DAGs with fixed K. We employed a lazy data generation procedure whereby node conditional probability distributions were generated only when they were required by the sampling, a technique which allows generation of data for arbitrarily dense networks.

In all cases we assume a uniform prior over non-forbidden structures and thus allow $p_s(X_i, \boldsymbol{P}_i) = 1$ for all i. We also adopted the K2 parameter prior [Cooper and Herskovits, 1992] which sets $\alpha_{ijk} = 1$ for all (i, j, k). This criterion has the property of weighting all local distributions of parameters uniformly. All variables in our synthetic tests were binary: $N_c = N_f = 2$, and for all experiments $N_{test} = 1000$. In many experiments we sampled the density of generating graphs K uniformly from a set $\{1, 2, \ldots, N\}$; we use the notation $K \hookleftarrow \{1, \ldots, N\}$ to denote this procedure.

In all experiments, π was chosen to be a fixed total ordering of the variables. At least three heuristics were used to generate π: (1) generate a random ordering, (2) generate two opposite random orderings and average predictions of each, and (3) use a topological sort of the graph obtained by GTT. These methods produced comparable results, but (2) and (3) performed a few percent better. In all experiments performed below, method (3) was used to generate π.

4.2 Experimental Results

The first experiment tested the degree of error incurred by model averaging over the class $\mathcal{L}_k^n(\pi \mid D)$ instead of the full class $\mathcal{L}_k(\pi)$. The degree of error was measured in terms of the percent difference in ROC areas, Δ, using $k = 1$, $N_r = 100$, $N_{trials} = 40$, N varied over the values $\{15, 20, 30, 40, 50, 60, 80, 100\}$, n varied over the values $\{5, 7, 10, 12, 15\}$, and $K \hookleftarrow [1, \ldots, N]$. The compiled results are shown in Table 2a–c. In this and

n	**Δ (%)**
1	15 ± 2.4
5	$.74 \pm .55$
7	$.24 \pm .42$
10	$.15 \pm .29$
12	$-.01 \pm .36$
15	$.12 \pm .14$

(a)

N	**Δ (%)**
20	$.023 \pm .067$
30	$.0016 \pm .18$
40	$.036 \pm .22$
60	$.10 \pm .22$
80	$.17 \pm .29$
100	$.23 \pm .32$

(b)

k	**Δ (%)**
1	$.032 \pm .09$
2	$.036 \pm .11$
3	$.093 \pm .23$
4	$.042 \pm .32$

(c)

Table 2: The percent difference (Δ) in ROC area between model averaging over $\mathcal{L}_k(\pi)$ and model averaging over $\mathcal{L}_k^n(\pi \mid D)$ as various parameters are varied, with the 99% confidence intervals,.

all subsequent tables, the error bars denote the 99% confidence interval of the mean.

Table 2a shows the dependence on n averaged over all values of N, showing that for an approximation level $n \gtrsim 10$ the difference in ROC area between the exact MA and AMA is less than 0.5% with confidence $P > 0.99$. Table 2b shows the dependence on N averaged over the values of $n \in \{10, 12, 15\}$. This table shows that with n set to reasonable values the approximation error is bounded under 0.6% with 99% confidence up to $N = 100$. Finally, one might expect the approximation error to increase as k was increased, since for a fixed n, $\mathcal{L}_k(\pi)$ includes increasingly more structures than $\mathcal{L}_k^n(\pi \mid D)$ as k increases. Table 2c shows the results of varying k in an experiment with $N = 20$, $N_r = 100$, $K \hookleftarrow [1, \ldots, N]$, $n = 12$ and k varied from 1 to 4. For all values of k the error in terms of the difference in ROC area is below 0.4% with 99% confidence.

Next, we generated synthetic data to test the performance of AMA relative to the other methods. For all tests, the approximation parameter was fixed to $n = 12$. We performed four tests, each varying one of the parameters in Table 1 while fixing the remaining parameters to particular values. These results are shown in Table 3.

Due to space constraints we only report here the results for AMA versus GTT. However, we note that both GTT and AMA achieved on average higher performance than SNN for almost every experimental run

N	δ (%)	Q_1	Q_3
5	16 ± 1.3	7.7	27
10	11 ± 1.5	4.0	23
20	7.1 ± 1.6	.14	17
40	12 ± 1.4	.75	23
80	14 ± 1.5	1.8	28
160	20 ± 4.1	.17	44

k	δ (%)	Q_1	Q_3
1	$.322 \pm .72$	-4.7	13
2	$9.05 \pm .46$	2.7	16
3	$10.5 \pm .37$	3.3	16
4	$10.9 \pm .38$	3.5	17
5	$10.9 \pm .43$	3.8	17
6	$10.2 \pm .45$	3.1	16

K	δ (%)	Q_1	Q_3
2	7.6 ± 1.7	.43	13
4	14 ± 2.2	3.6	23
8	11 ± 2.1	2.4	18
16	7.2 ± 1.9	-.40	13
32	5.3 ± 1.8	-1.1	7.9
50	6.0 ± 1.6	-.50	9.7

N_r	δ (%)	Q_1	Q_3
25	9.4 ± 2.7	1.2	13
50	9.4 ± 2.5	1.6	16
100	11 ± 2.3	4.0	16
400	10 ± 2.2	3.6	16
800	5.5 ± 3.0	-.30	13
3200	-7.4 ± 5.2	-26	13

Table 3: The performance of AMA versus GTT as several parameters are varied. The error terms indicate the 99% significance level. Q_1 and Q_3 denote the first and third quartiles, respectively.

performed. NMA performed comparably to AMA for small N_r, but performed worse than both GTT and AMA for large N_r. Later (Table 5) we present quantitative comparisons of all four techniques on real-world data.

The top-left quadrant of Table 3 shows the results of varying the number of nodes N, while holding $N_r = 100$, $k = 2$, and with $K \hookleftarrow [1, \ldots, N]$. For all values of N using this configuration of parameters, AMA outperformed GTT at the 99% significance level, the difference generally increasing as N grew very large or very small. Probably the minimum in this curve around $N = 20$ reflects the tradeoff that as N gets very small, bounding the in-degree of the networks to $k = 2$ comes closer and closer to the limiting case of $k = N$; whereas, generally as the ratio N/N_r increases we expect model averaging to benefit over a single model.

In the bottom-left quadrant of Table 3, we varied the density of generating graphs K while holding $N_r = 100$, $k = 2$ and $N = 50$. Again, for this set of measurements, AMA always outperformed GTT at the 99% significance level. The results were most apparent when $K \lesssim 10$, an encouraging result since it is a common belief that most real-world networks are sparse.

In the top-right of Table 3, we varied the maximum number of parents allowed in $\mathcal{L}_k$ from $1 \leq k \leq 6$, while holding $N_r = 100$, $N = 20$, and $K \hookleftarrow [1, \ldots, N]$. The value of δ is surprisingly insensitive to the value of k for $k \geq 2$. Finally, in the bottom-right, we varied the number of records N_r while fixing $N = 20$, $k = 3$ and $K \hookleftarrow [1, \ldots, N]$. As expected, for smaller values of N_r model averaging performs well versus GTT. As N_r grows GTT eventually outperforms AMA at the 99% significance level. It was observed in other experiments that the ability of GTT to outperform AMA at high N_r depended strongly on the value of k. When it was computationally feasible to set $k \simeq N$, AMA typically would outperform GTT even at high N_r.

The performance of AMA was also tested by generating training and test data with the benchmark ALARM network. In this case, $N = 36$ and $K = 4$ were fixed by the network, and a test was performed with $k = 3$, $n = 10$, and N_r systematically varied. The results in Table 4 are shown for classification on the *kinked tube* and *anaphylaxis* nodes. These results are of interest because they demonstrate that the qualitative performance of the AMA classifier depends not just on global network features but also on features specific to the classification node.

N_r	δ^{kt} (%)	Q_1^{kt}	Q_3^{kt}	δ^{an} (%)	Q_1^{an}	Q_3^{an}
50	32 ± 13	24	55	2.6 ± 3.2	-9.2	17
100	23 ± 11	8.8	53	$.66 \pm 3.3$	-11	16
200	13 ± 9.0	-1.2	32	-2.6 ± 4.2	-17	16
400	12 ± 6.9	-1.2	34	-3.3 ± 5.4	-21	18
800	3.7 ± 6.9	-8.5	23	2.0 ± 5.0	-11	21
3200	$-.07 \pm 14$	-19	15	5.6 ± 7.1	-7.5	19

Table 4: AMA performance v.s. GTT on synthetic data generated using the ALARM network and classifying on *kinked tube* (kt) and *anaphylaxis* (an).

Finally, we performed measurements of the performance of AMA, SNN, GTT and NMA on 21 data sets taken from the UCI online database [Blake and Merz, 1998]. These results are shown in Table 5. Here the score δ_d^i for classifier C_i was calculated according to Procedure 2, where $M_2 = C_i$ and M_1 was taken to be the maximum scoring classifier for the data set d. For example, in the monks-2 database, AMA was the highest scoring classifier and covered 48% of the remaining area for SNN and GTT and 21% of the remaining area for NMA. The ROC area will in general depend on which state of the classification variable is considered to be the "positive" state; therefore, the scores in Table 5 are average scores for all ROC curves associated with a particular classification variable; therefore some data sets (e.g., wine) have no zero entries when two or more classifiers score highest on different curves.

We have underlined the top two scoring classifiers for each data set to emphasize the fact that AMA was typi-

Data set	δ^{SNN}	δ^{GTT}	δ^{NMA}	δ^{AMA}	N	k	N_r
haberman	0.35	0.35	**0**	**0**	4	4	306
hayes-roth	0.32	0.32	**0**	**0.01**	6	6	132
monks-3	0.83	**0.24**	0.82	**0**	7	7	552
monks-1	0.98	**0**	0.98	**0**	7	7	554
monks-2	0.48	0.48	**0.21**	**0**	7	7	600
chess krk	0.54	**0**	0.54	**0.32**	7	7	28055
ecoli	0.03	**0.01**	0.02	**0**	8	8	336
yeast	**0.04**	0.11	**0.04**	0.07	8	8	1484
abalone	0.12	0.08	**0.05**	**0**	9	9	4176
cpu-perf	0.13	0.31	**0.01**	**0.11**	10	10	209
glass	**0.10**	**0.04**	0.15	0.13	10	10	214
cmc	**0.01**	0.07	**0.01**	0.04	10	10	1473
sol-flare-C	0.03	0.09	**0.02**	**0.01**	11	11	322
sol-flare-M	**0**	0.44	**0.17**	0.20	11	11	322
sol-flare-X	**0.06**	**0.01**	0.18	0.33	11	11	322
wine	0.14	**0.01**	0.16	**0.06**	14	7	177
credit-scrn	**0**	0.12	0.09	**0.02**	16	5	652
letter-rec	0.38	**0**	0.38	**0.01**	17	5	20000
thyroid	0.17	0.28	**0**	**0.11**	21	5	7200
brst-canc-w	0.28	**0**	0.29	**0.23**	32	3	569
connect-4	**0.49**	**0**	**0.49**	0.52	43	2	67557

Table 5: Experimental results for 21 UCI data sets.

cally more robust on these data sets than the other classifiers, scoring in the top two 15/21 times compared to 7/21, 10/21, and 11/21 for SNN, GTT and NMA, respectively. The average difference Δ^i between classifier i and AMA: $\Delta^i \equiv \frac{1}{21}\sum_d(\delta_d^{AMA} - \delta_d^i)$, was calculated to gauge the statistical significance of these experiments. The results were: $\Delta^{SNN} = 15.8 \pm 6.7\%$ (significant at the 99% level), $\Delta^{GTT} = 3.7 \pm 5.3\%$ (significant only at the 75% level), and $\Delta^{NMA} = 11.7 \pm 6.3$ (significant at the 95% level). These results are promising, but need to be extended to further investigate the performance of AMA.

5 Discussion

We have shown that it is possible to construct a single DAG model that will perform linear-time approximate model averaging over the $\mathcal{L}_k^n(\pi \mid D)$ class of models. We have demonstrated empirically that even with relatively little effort in choosing a good partial ordering π, classifications obtained by model averaging over $\mathcal{L}_k^n(\pi \mid D)$ can be beneficial compared to other BN classifiers. The benefits of AMA were not without cost; although comprehensive measurements were not taken, it was observed that the initial model construction time typically was 3-10 times longer for AMA than it took for the greedy search to converge for the values of k and n used in our experiments.

The AMA technique is interesting because of its simplicity of implementation. Existing systems that use Bayesian network classifiers can trivially be adapted to use model averaging by replacing their existing model with a single summary model.

Future work includes finding a better method for optimizing the ordering π, possibly by doing a search over orderings as in [Friedman and Koller, 2000]. Also, our experimental results should be expanded to more extensively characterize the performance of AMA for classification on real-world data. Another possible extension is to relax the assumption of complete data, possibly by using the EM algorithm or MCMC sampling to estimate Equation 14 from data.

6 Acknowledgements

This work was supported in part by grant number S99-GSRP-085 from the National Aeronautics and Space Administration under the Graduate Students Research Program, by grant IIS-9812021 from the National Science Foundation and by grants LM06696 from the National Library of Medicine.

References

[Blake and Merz, 1998] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.

[Buntine, 1991] W. Buntine. Theory refinement on Bayesian networks. In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence (UAI–91)*, pages 52–60, San Mateo, California, 1991. Morgan Kaufmann Publishers.

[Cooper and Herskovits, 1992] Gregory F. Cooper and Edward Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4):309–347, 1992.

[Dash and Cooper, 2002] Denver Dash and Gregory F. Cooper. Exact model averaging with naive Bayesian classifiers. *Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002), to appear*, 2002.

[Domingos and Pazzani, 1997] Pedro Domingos and Michael Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.

[Friedman and Koller, 2000] Nir Friedman and Daphne Koller. Being Bayesian about network structure. In *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*, pages 201–210, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[Friedman *et al.*, 1997] Nir Friedman, Dan Geiger, Moises Goldszmidt, G. Provan, P. Langley, , and P. Smyth. Bayesian network classifiers. *Machine Learning*, 29:131, 1997.

[Heckerman *et al.*, 1995] David Heckerman, Dan Geiger, and David M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:197–243, 1995.

[Madigan and Raftery, 1994] David Madigan and Adrian E. Raftery. Model selection and accounting for model uncertainty in graphical models using Occam's window. *Journal of the American Statistical Association*, 89:1535–1546, 1994.

[Madigan and York, 1995] David Madigan and J. York. Bayesian graphical models for discrete data. *International Statistical Review*, 63:215–232, 1995.

[Meila and Jaakkola, 2000] Marina Meila and Tommi S. Jaakkola. Tractable Bayesian learning of tree belief networks. In *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*, pages 380–388, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[Spirtes *et al.*, 1993] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. Springer Verlag, New York, 1993.

[Verma and Pearl, 1991] T.S. Verma and Judea Pearl. Equivalence and synthesis of causal models. In *Uncertainty in Artificial Intelligence 6*, pages 255 –269. Elsevier Science Publishing Company, Inc., New York, N. Y., 1991.

[Volinsky, 1997] C.T. Volinsky. *Bayesian Model Averaging for Censored Survival Models*. PhD dissertation, University of Washington, 1997.