Journal of Machine Learning Research 5 (2004) 1177–1203 Submitted 11/03; Revised 9/04; Published 9/04

Model Averaging for Prediction with Discrete Bayesian Networks

Denver Dash DENVER.H.DASH@INTEL.COM
Intel Research
SC12-303
3600 Juliette Lane
Santa Clara, CA 95054, USA

Gregory F. Cooper GFC@CBMI.UPMC.EDU
Center for Biomedical Informatics
University of Pittsburgh
Pittsburgh, PA 15260, USA

Editor: David Madigan

Abstract

In this paper[1] we consider the problem of performing Bayesian 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, and yet model averaging over these structures can be performed using $O(\binom{N}{k} \cdot N)$ operations. Furthermore we show that there exists a single Bayesian network that defines a joint distribution over the variables that is equivalent to model averaging over these structures. Although constructing this network is computationally prohibitive, we show that it can be approximated by a tractable network, allowing approximate model-averaged probability calculations to be performed in $O(N)$ time. Our result also leads to an exact and linear-time solution to the problem of averaging over the 2^N possible feature sets in a naïve Bayes model, providing an exact Bayesian solution to the troublesome feature-selection problem for naïve Bayes classifiers. We demonstrate the utility of these techniques in the context of supervised classification, showing empirically that model averaging consistently beats other generative Bayesian-network-based models, even when the generating model is not guaranteed to be a member of the class being averaged over. We characterize the performance over several parameters on simulated and real-world data.

Keywords: Bayesian networks, Bayesian model averaging, classification, naïve Bayes classifiers, feature selection

1. Introduction

A probabilistic model M over a set of variables X, is a parameterization of the joint distribution $P(X)$ over X. There are many practical uses for $P(X)$, including the ability to calculate expectations, $E(X)$, of configurations of variables, the ability to calculate the *most likely explanation* of some observed evidence, the ability to *update beliefs* about some variables given some other variables, $P(X_i, X_j \mid X')$, etc. In short virtually any probabilistic quantity of interest involvoing the variables X can be calculated once $P(X)$ is known. Bayesian networks (BNs) (cf., Pearl, 1988) are a popular

1. This is a combined and expanded version of previous conference and workshop papers (Dash and Cooper, 2002, 2003).

class of graphical probabilistic models that allow $P(X)$ to be specified in practice even when $|X|$ is very large, by explicating independence between the variables X.

Many algorithms for learning BNs from data (Verma and Pearl, 1991; Cooper and Herskovits, 1992; Spirtes, Glymour, and Scheines, 1993; Heckerman, Geiger, and Chickering, 1995; Friedman, Geiger, and Goldszmidt, 1997) have been used effectively to learn the structure of a BN model from data, typically by performing a search over structures using the posterior probability, $P(S \mid D)$, of the structure given the data as a measure of quality. While learning a particular BN structure has shown to be useful, it suffers from the fact that a single model makes strong independence assumptions among the variables of interest that may not be true, or may only be approximately true in reality. That is, the process of learning a single network affords no way of capturing the uncertainty in the model structure. The most principled alternative to selecting a particular network structure, is to calculate the full joint posterior $P(X \mid D)$ by averaging over all possible BN structures. 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.

One especially common use for learning the joint distribution from data is *supervised classification*. 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 $F = \{F_1, F_2, \dots, F_N\}$ into one of various classes of interest. A probabilistic model accomplishes this goal by calculating the posterior probability, $P(C \mid F)$, of the class given the features. One of the simplest probabilistic classifiers for this task is the naïve classifier (cf., Duda and Hart, 1973), which, without inferring any structural information from the database, can still perform surprisingly well at the classification task (Domingos and Pazzani, 1997; Friedman, 1997; Ng and Jordan, 2002). Classification using a single Bayesian network model and no missing feature-vector data can be performed in $O(N)$ time. When the feature vector is incomplete then standard algorithms (e.g., Lauritzen and Spiegelhalter, 1988) for Bayesian network inference can be used for classification. The drawbacks of selecting a single model for classification manifests themselves as over-fitting of the data, leading to poor classification accuracy on future data sets; however, model averaging has been shown to reduce over-fitting and provide better generalization (Madigan and Raftery, 1994).

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 directed acyclic graphs (DAGs). 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.

There has been other work on making model averaging over Bayesian network structures efficient: Methods for approximate MA classification using both selective pruning (Madigan and Raftery, 1994; Volinsky, 1997) and Monte-Carlo techniques (Madigan and York, 1995) exist and have been shown to improve performance in prediction tasks; however these methods are approximate, and do not have the complexity guarantees that our method possesses. Friedman and Koller (2003) 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 credit to Buntine (1991), that we extend in order to prove our key theoretical result. We discuss this issue in detail in Section 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 prediction. Other work has been done (Meila and Jaakkola, 2000; Cerquides and de Màntaras, 2003) on performing exact model averaging for prediction over all tree-structures in $O(N^3)$ time. This research also uses similar assumptions and similar decompositions used by Friedman and Koller (2003) and in this paper. 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.

The primary contributions in this paper are as follows: (1) we extend the decomposition of Buntine (1991) to apply to the task of prediction, (2) we show that MA calculations over this class can be reproduced by a single network structure S^* which, if it can be constructed tractably, thereafter allows approximate MA predictions to be performed using standard Bayesian network inference, (3) we show that, for the class of naïve models, calculation of S^* (including parameters) can be performed in linear time in the number of variables, and we demonstrate empirically that model averaging over naïve classifiers improves performance, and (4) we develop a technique to make model averaging practical for arbitrary orderings, and we demonstrate empirically that this technique can result in improved classification (compared to other Bayesian network classifiers) even when no ordering information is known *a priori*.

Aside from the practical issue of achieving accurate predictions, our technique is interesting from an analytical perspective. As an example, recently, Domingos (2000) made an argument based on empirical and theoretical grounds that Bayesian model averaging can actually exacerbate the over-fitting problem in machine learning. Empirically, he shows that rule-learners that approximate pure Bayesian model averaging closer and closer achieve successively higher error rates than a rule-learner that uses the more *ad hoc* technique of bagging. He explains this observation as a consequence of the likelihood's exponential sensitivity to random fluctuations in the data, and surmises that the effect will be significant even for small data sets and will be amplified as the number of models being averaged over increases. Our experiments here present a direct test of this assertion, obtaining results that conflict with the conclusions of Domingos.

In Section 2 we formally frame the problem and state our assumptions and notation, and re-derive previous results. 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. Previous Results

In this section we frame the problem, introduce our notation and review relevant previous research, re-deriving the results of Friedman and Koller (2003) and Buntine (1991) and casting them into notation that will allow us to extend them for prediction in Section 3.

2.1 Assumptions and Notation

The general supervised classification problem can be framed as follows: Given a set of features $F = \{F_1, F_2, \dots, F_N\}$, a set of classes $\mathcal{C} = \{C_1, C_2, \dots C_{N_c}\}$, and a labelled training data set $D = \{D_1, D_2, \dots, D_{N_D}\}$ generated from some distribution P, construct a model to predict into which class future feature vectors sampled from P are most likely to reside. We use the notation X_i to refer to the nodes when we need to have a uniform notation; we use the convention that $X_i \equiv F_i$ and $X_0 \equiv C$, and we use X to denote the collective set of nodes in the network. A directed graph $G(X)$ is defined as a pair $\langle X, E \rangle$, where E is a set of directed edges $X_i \rightarrow X_j$, such that $X_i, X_j \in X$. If X is a random variable, we let $\mathit{Rng}(X)$ denote the range of X. We typically use boldface symbols to denote sets and non-boldface type to denote elements of sets, when possible.

We are considering the problem of averaging over the space of *Bayesian network* structures. For a set of variables X, a Bayesian network on X is a graphical model which factorizes the joint distribution $P(X)$ over X. In particular:

Definition 1 (Bayesian network) *Given a set X of N variables, a Bayesian network B on X is a pair $B = \langle S, \theta \rangle$, where $S = \langle X, E \rangle$ is a directed acyclic graph over X, and $\theta = \{\theta_0, \theta_1, \ldots, \theta_N\}$ are the parameters of the network that represent the set of conditional probability distributions for each variable in X given its parents in S.*

We make the following assumptions:

Assumption 1 (Multinomial variables) *Each node X_i represents a discrete random variable with r_i possible states: $\mathit{Rng}(X_i) = \{x_i^1, x_i^2, \ldots, x_i^{r_i}\}$.*

We use Pa_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 (defining q_i=1 if $Pa_i = \emptyset$), which we enumerate as: $\mathit{Rng}(Pa_i) = \{p_i^1, p_i^2, \ldots, p_i^{q_i}\}$; for example, if X_i has 3 binary parents then $q_i = 8$.

Under the assumption of multinomial variables, a conditional probability distribution θ_i for variable X_i will take the form of a conditional probability table (CPT) with components $\theta_{ijk} = P(X_i = x_i^k \mid Pa_i = p_i^j)$, and for a fixed network structure S, the components θ_{ijk} form the parameters of the Bayesian network model and define the joint distribution over all variables assuming the Markov condition holds. We use the symbol θ_{ij} to denote the entire conditional probability distribution function for the i-th node and the j-th parent configuration, and the symbol θ 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 use the shorthand that if Q_{ijk} is some quantity associated with coordinates (ijk), then $Q_{ij} \equiv \sum_k Q_{ijk}$.

Assumption 2 (Complete labelled training data) *The training data set D contains no record $D_l \in D$ such that D_l has a non-observed component.*

We will discuss ways to relax this assumption in Section 5. We let N_{ijk} denote the sufficient statistics of the data set (i.e., the number of times that node X_i achieved state k when parent set Pa_i was in the j-th configuration).

Assumption 3 (Dirichlet priors) *The prior beliefs over parameter values are given by a Dirichlet distribution.*

We let α_{ijk} denote the Dirichlet hyperparameter corresponding to the network parameter θ_{ijk}. For simplicity, we assume α_{ijk} can be calculated in $O(1)$ time and space; this is the case, for example, with two popular metrics, the K2 metric (Cooper and Herskovits, 1992) and the BDeu metric (Heckerman, Geiger, and Chickering, 1995).

Assumption 4 (Parameter independence) *For any given network structure S, each probability distribution θ_{ij} is independent of any other probability distribution $\theta_{i'j'}$:*

$$P(\theta \mid S) = \prod_{i=0}^{N} \prod_{j=1}^{q_i} P(\theta_{ij} \mid S). \tag{1}$$

Finally, we take the assumption that the priors on parameters θ_{ijk} for a node X_i depend only on the local structure. This assumption is known as *parameter modularity* (Heckerman, Geiger, and Chickering, 1995):

Definition 2 (Parameter modularity) *Let X be a set of variables with $X_i \in X$. For any two network structures S_1 and S_2 over X, if $Pa_i|_{S_1} = Pa_i|_{S_2}$ then $P(\theta_{ijk} \mid S_1) = P(\theta_{ijk} \mid S_2)$.*

2.2 Averaging Over Parameters with a Fixed Network Structure

One common goal in machine learning with Bayesian networks is to calculate the probability of a configuration $X = x$ of a set of variables X. This can be used for predicting likely configurations of variables, or it can be queried for any conditional probability of interest over the variables in X (e.g., $P(X_1, X_2 \mid X_3)$) which could be useful for prediction. For a fixed network structure S and a fixed set of network parameters θ, $P(X = x \mid S, \theta)$ can be calculated in $O(N)$ time:

$$P(X = x \mid S, \theta) = \prod_{i=0}^{N} \theta_{iJK}, \tag{2}$$

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

When, rather than a fixed set of parameters, a database D is given, it is necessary to average over all possible configurations of the parameters θ:

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

where the second line follows from Equation 2. 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, Geiger, and Chickering, 1995):

$$P(X = x \mid S, D) = \prod_{i=0}^{N} \frac{\alpha_{iJK} + N_{iJK}}{\alpha_{iJ} + N_{iJ}}, \tag{3}$$

where we have used the notation: $\alpha_{ij} = \sum_k \alpha_{ijk}$. Comparing this result to Equation 2 illustrates the well-known result that a single network with a fixed set of parameters $\tilde{\theta}$ given by

$$\tilde{\theta}_{ijk} = \frac{\alpha_{ijk} + N_{ijk}}{\alpha_{ij} + N_{ij}} \tag{4}$$

will produce predictions equivalent to those obtained by averaging over all parameter configurations. We refer to Equation 4 as the *standard parameterization*.

2.3 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, for example, the posterior probability $P(X_L \rightarrow X_M \mid D)$

of a particular arc $X_L \rightarrow X_M$ averaged over all in-degree-bounded networks consistent with a fixed ordering. Friedman and Koller then showed how this quantity could be used in a MCMC search for the most likely structural feature, where the search went over orderings instead of DAGs. One might be interested in this quantity, for example, if you were interested in a Bayesian estimate for the structural dependency relations of the system given the data; see Friedman and Koller (2003) for more motivation for why this quantity would be useful. In this section we re-derive the result of Buntine.

The derivation required the ability to calculate efficiently the prior probability $P(S)$ that a given structure S generated the database D. An additional assumption was introduced, labelled *structure modularity* by Friedman and Koller:

Assumption 5 (Structure modularity) *The prior of a structure S, $P(S)$, can be factored according to the network*

$$P(S) \propto \prod_{i=0}^{N} p_s(X_i, Pa_i), \tag{5}$$

where $p_s(X_i, Pa_i)$ is some function that depends only on the local structure (X_i and Pa_i).

Any metric that assesses the structure prior $P(S)$ based on a difference in arcs between S and some prior network structure S' (as suggested by Heckerman, Geiger, and Chickering (1995)) will satisfy this condition. Also the uniform distribution will obviously satisfy this assumption, and requires $O(1)$ time to assess. Obviously, we restrict ourselves to structures that give probability zero to a non-acyclic DAG.

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

$$\begin{aligned} P(X_L \rightarrow X_M \mid D) = \\ c \sum_S \delta_K(X_L \rightarrow X_M \in S) \cdot P(D \mid S) \cdot P(S), \end{aligned} \tag{6}$$

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

$$\delta_K(Z) = \begin{cases} 1 & \text{if } Z = \textit{true} \\ 0 & \text{otherwise.} \end{cases}$$

The summation in Equation 6 includes a super-exponential number of network structures, and therefore appears to be intractable. Buntine handled this problem by imposing a total ordering on the nodes and restricting the maximum number of parents, k, that each node can have. Generalizing his results to a partial ordering π instead of a total ordering is straightforward, and we do that here. For a given partial ordering π and a particular node X_i, it is required 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 $\pi = \langle \{X_1, X_3\}, \{X_2, X_4\} \rangle$ and a maximum in-degree $k = 2$ would yield the following enumeration of parent sets for X_2: $\{Pa_2^0 = \emptyset, Pa_2^1 = \{X_1\}, Pa_2^2 = \{X_3\}, Pa_2^3 = \{X_1, X_3\}\}$.

The class of models consistent with π with bounded in-degree of k we denote as $\mathcal{L}_k^\pi$:

Definition 3 ($\mathcal{L}_k^\pi$) *For a given integer $k \leq N$ and a given partial ordering π of X, a DAG $G = \langle X, E \rangle \in \mathcal{L}_k^\pi$ iff arcs are directed from higher to lower levels and no variable has more than k parents: $X_i \rightarrow X_j \in E \Rightarrow \pi(X_i) > \pi(X_j)$, and $X_i \in X \Rightarrow |Pa_i| \leq k$.*

Note that if $D_1, D_2 \in \mathcal{L}_k^\pi$ and $D_1 \neq D_2$ then D_1 is statistically distinguishable from D_2 because it will include a different set of adjacencies (Verma and Pearl, 1991); so when averaging over the class $\mathcal{L}_k^\pi$, we are never averaging over equivalent DAGs.

The number of structures in $\mathcal{L}_k^\pi$ is still exponentially large in the worst-case; each node at level l can choose up to k parents from among all the nodes in levels $l' > l$. For $k \leq N/2$, $\mathcal{L}_k^\pi$ in the worst-case includes at least $\Omega\left[\binom{N/2}{k}^{N/2}\right]$ network structures. This result corresponds to the case where π consists of two levels, each with $N/2$ nodes; each of the $N/2$ nodes in the bottom level can therefore choose up to $\binom{N/2}{k}$ possible parents.[2]

Given the assumptions of this paper, $P(D \mid S)$, can be written just in terms of hyperparameters and sufficient statistics (Cooper and Herskovits, 1992; Heckerman, Geiger, and Chickering, 1995):

$$P(D \mid S) = \prod_{i=0}^{N} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}. \tag{7}$$

Given structure modularity (Assumption 5) and Equation 7, Equation 6 can be written as

$$P(X_L \to X_M \mid D) = c \sum_S \prod_{i=0}^{N} \rho_{iLM}^S. \tag{8}$$

The ρ_{iLM}^S functions are given by

$$\rho_{iLM}^S = \delta_K[M \neq i \vee X_L \in Pa_i] \cdot p_s(X_i, Pa_i) \cdot \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})}, \tag{9}$$

and can be calculated using information that depends only on nodes X_i and Pa_i.

Even restricting structures to those in a particular $\mathcal{L}_k^\pi$ class, as already mentioned, the summation in Equation 8 still contains an exponential number of structures in the worst-case. However, the following theorem (similar to one from Buntine (1991)) shows that $P(X_L \to X_M \mid D, \mathcal{L}_k^\pi)$ can be calculated with relative efficiency:

Theorem 1 *Assuming $S \in \mathcal{L}_k^\pi$, equation 8 can be written as $P(X_L \to X_M \mid D, \mathcal{L}_k^\pi) = c \prod_{i=0}^{N} \sum_{\nu=0}^{\mu_i} \rho_{iLM}^\nu$, where ρ_{iLM}^ν denotes ρ_{iLM}^S for the νth parent set Pa_i^ν of X_i and the summation goes over all parent sets available to node X_i under the restrictions of $\mathcal{L}_k^\pi$.*

Proof: For notational simplicity, we will drop the LM subscripts on the ρ functions: $\rho_i^\nu \equiv \rho_{iLM}^\nu$. Expanding the sum in Equation 8 yields

2. We are not asserting that this two-level ordering is the absolute worst-case, only that the worst-case must have at least this many network structures.

$$P(X_L \to X_M \mid D, \mathcal{L}_k^\pi) \propto \left.\begin{array}{cl} & \rho_0^0 \cdot \rho_1^0 \cdot \ldots \cdot \rho_N^0 \\ + & \rho_0^1 \cdot \rho_1^0 \cdot \ldots \cdot \rho_N^0 \\ \vdots & \qquad \vdots \\ + & \rho_0^{\mu_0} \cdot \rho_1^0 \cdot \ldots \cdot \rho_N^0 \\ & \\ + & \rho_0^0 \cdot \rho_1^1 \cdot \ldots \cdot \rho_N^0 \\ + & \rho_0^1 \cdot \rho_1^1 \cdot \ldots \cdot \rho_N^0 \\ \vdots & \qquad \vdots \\ + & \rho_0^{\mu_0} \cdot \rho_1^1 \cdot \ldots \cdot \rho_N^0 \\ \vdots & \qquad \vdots \\ + & \rho_0^0 \cdot \rho_1^0 \cdot \ldots \cdot \rho_N^1 \\ + & \rho_0^1 \cdot \rho_1^0 \cdot \ldots \cdot \rho_N^1 \\ \vdots & \qquad \vdots \\ + & \rho_0^{\mu_0} \cdot \rho_1^0 \cdot \ldots \cdot \rho_N^1 \\ \vdots & \qquad \vdots \\ + & \rho_0^{\mu_0} \cdot \rho_1^{\mu_1} \cdot \ldots \cdot \rho_N^{\mu_N} \end{array}\right\} \Omega\left[\binom{N/2}{k}^{N/2}\right] \text{ terms, worst-case.}$$

We define the symbol Σ_m to denote the structure sum of the product up to and including the m-th node:

$$\begin{array}{rcl} \Sigma_m & \equiv & \rho_0^0 \cdot \rho_1^0 \cdot \ldots \cdot \rho_m^0 \\ & + & \rho_0^1 \cdot \rho_1^0 \cdot \ldots \cdot \rho_m^0 \\ & \vdots & \qquad \vdots \\ & + & \rho_0^{\mu_0} \cdot \rho_1^{\mu_1} \cdot \ldots \cdot \rho_m^{\mu_m}. \end{array}$$

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

$$\Sigma_i = \Sigma_{i-1} \cdot \sum_{\nu=1}^{\mu_i} \rho_i^\nu, \qquad \Sigma_{-1} = 1.$$

Finally, expanding the recurrence relation yields the expression for $P(X_L \to X_M \mid D, \mathcal{L}_k^\pi)$:

$$P(X_L \to X_M \mid D, \mathcal{L}_k^\pi) = c \prod_{i=0}^{N} \sum_{\nu=1}^{\mu_i} \rho_i^\nu. \tag{10}$$

$\square$

Thus, a summation of $\Omega\left[\binom{N/2}{k}^{N/2}\right]$ terms can be performed in $O\left[\binom{N}{k} \cdot N\right]$ operations.

3. Model Averaging for Prediction

In this section we show how to extend the results of Section 2 to efficiently calculate the quantity $P(X = x \mid D, \mathcal{L}_k^\pi)$ averaged over the class $\mathcal{L}_k^\pi$.

This quantity can be written as

$$P(X = x \mid D, \mathcal{L}_k^\pi) = \frac{1}{P(D)} \sum_{S \in \mathcal{L}_k^\pi} \prod_{i=0}^{N} \tilde{\theta}_{iJK} \cdot P(D \mid S) \cdot P(S), \tag{11}$$

where $\tilde{\theta}_{iJK}$ are the standard parameters given in Equation 4. Given structure modularity and Equation 7, Equation 11 can be written in a form very similar to Equation 8:

$$P(X = x \mid D, \mathcal{L}_k^\pi) = c \sum_{S \in \mathcal{L}_k^\pi} \prod_{i=0}^{N} \tilde{\rho}_{iJ_{\mathbf{x}}^S K_{\mathbf{x}}}, \tag{12}$$

where here the $\tilde{\rho}_{iJ_{\mathbf{x}}^S K_{\mathbf{x}}}$ functions are given by

$$\tilde{\rho}_{iJ_{\mathbf{x}}^S K_{\mathbf{x}}} = \tilde{\theta}_{iJ_{\mathbf{x}}^S K_{\mathbf{x}}} \cdot \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \cdot \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ijk})} \cdot p_s(X_i, Pa_i), \tag{13}$$

and c is a constant (not dependent on S) equal to $1/P(D)$. We subscript the indices J and K from Equation 2 with an x to indicate that they are fixed by a particular configuration of X, and J is indexed by S to emphasize that the value of the parent configuration index depends on the number of parents and therefore the structure S of the network. Although this notation may seem cumbersome, we believe it clarifies the analysis later.

As in Section 2.3, the worst-case number of terms in the summation of Equation 12 is exponential in the number of features N. The $\tilde{\rho}_{iJ_{\mathbf{x}}^S K_{\mathbf{x}}}$ functions again can be calculated using only information local to node X_i and Pa_i. Following a derivation identical to that for averaging $P(X_L \to X_M \mid D, \mathcal{L}_k^\pi)$ in Section 2.3, yields the following :

$$P(X = x \mid D, \mathcal{L}_k^\pi) = c \prod_{i=0}^{N} \sum_{\nu=1}^{\mu_i} \tilde{\rho}^{\nu}_{iJ_{\mathbf{x}}^{\nu} K_{\mathbf{x}}}. \tag{14}$$

Here the S index has been replaced with a ν indicating which parent set for node X_i is being considered. The following theorem shows that this summation can be performed in $O(\binom{N}{k} \cdot N \cdot k \cdot N_D \cdot R)$ time and $O(k \cdot N_D)$ space.

Theorem 2 *For N variables with a maximum number of states per variable given by R and a database of N_D records, Equation 14 can be calculated in $O(\binom{N}{k} \cdot N \cdot k \cdot N_D \cdot R)$ time and using $O(k \cdot N_D)$ space.*

Proof: The right-hand-side of Equation 14 includes N products and $\binom{N}{k}$ sums. Each ρ^{ν}_{iLM} term can be calculated in $O(k \cdot N_D \cdot R)$ time and $O(k \cdot N_D)$ space. This result follows because all sufficient statistics for a given node X_i can be stored in a tree of depth $O(k)$ and width $O(N_D)$, with the leaves of the tree holding the sufficient statistic for the given configuration of X_i and $Pa(X_i)$. To fill the tree requires N_D passes of the tree, thus taking $O(k \cdot N_D)$ time. Once the tree is constructed, it can be queried for any statistic in $O(k)$ time.

The number of possible parent configurations present in the data are bound by the number of records N_D; thus the number of (i, j) configurations for which $N_{ij} \neq 0$ is $O(N_D)$. Furthermore, all terms in the product of Equation 14 for which $N_{ij} = 0$ will equal 1 so will not contribute to the product. Thus, the calculation of products in Equation 13 require $O(k \cdot N_D \cdot R)$ time.

Putting all the steps together yields the claim of the theorem. $\square$

Although this result is relatively efficient, for classification purposes it may still be too complex of a calculation. The functional form of the above solution allows us to prove that exact model averaging over $\mathcal{L}_k^\pi$ can actually be performed with a single Bayesian network structure:

Theorem 3 *There exists a single Bayesian network model $M^* = \langle S^*, \theta^* \rangle$ that will define a joint distribution $P(X = x \mid S^*, \theta^*)$ that is equivalent to that 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 $Pa_i^* = \bigcup_{\nu=1}^{\mu_i} Pa_i^\nu$, and let θ^* be defined by

$$\theta^*_{ijk} = c^{1/N} \sum_{\nu=1}^{\mu_i} \tilde{\rho}^\nu_{iJ_j^\nu k}, \tag{15}$$

where the 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 considering a particular coordinate (ijk). It can be seen by direct comparison that substituting θ^*_{ijk} into Equation 2 will yield Equation 14. □

If we define functions $f(X_i, Pa_i^\nu \mid D)$ such that $f(X_i, Pa_i^\nu \mid D) = \tilde{\rho}^\nu_{iJ_j^\nu k} / \tilde{\theta}^\nu_{iJ_j^\nu k}$, then Equation 15 can be written as

$$\theta^*_{ijk} = c^{1/N} \sum_{\nu=1}^{\mu_i} \tilde{\theta}^\nu_{iJ_j^\nu k} \cdot f(X_i, Pa_i^\nu \mid D). \tag{16}$$

The functions $f(X_i, Pa_i^\nu \mid D)$ do not depend on the indices J_j^ν and k, and they can be interpreted as the local posterior probability that the parent set of X_i is in fact Pa_i^ν. Equation 16 thus provides the interpretation that M^* represents a structure-based smoothing where each standard parameter $\tilde{\theta}^\nu_{ijk}$ is weighted based on the likelihood that Pa_i^ν is the parent set of X_i. Since θ^*_{ijk} is interpretable as a probability, the constant $c^{1/N}$ serves as a normalization constant and need not be calculated directly.

There are some numerical complexities with calculating the parameters using Equation 16. One must essentially calculate quantities of the form

$$\theta^*_i = c^{1/N} \sum_\nu \tilde{\theta}^\nu_i \cdot \exp lf_i^\nu, \tag{17}$$

where $lf_i^\nu = \log f_i^\nu$ is a negative number with large absolute value. Exponentiating this value will usually be truncated to zero using floating-point arithmetic. In reality however, the normalization constant $c^{1/N}$ is also very small and in fact makes θ^*_i nonzero in many cases. To get around this problem, we use a known trick of shifting the exponentials so the largest term is equal to 1. The net result of this is to change the normalization constant, which is never calculated explicitly, i.e.

$$c^{1/N} \sum_\nu \tilde{\theta}^\nu_i \cdot \exp lf_i^\nu = \frac{c^{1/N}}{\exp(-lf_{max})} \sum_\nu \tilde{\theta}^\nu_i \cdot \exp(lf_i^\nu - lf_{max}), \tag{18}$$

Theorem 3 implies that, rather than performing the $O(\binom{N}{k} \cdot N)$ summation in Equation 14 for each case to be classified, in principle we need only construct a single model M^* and use standard Bayesian network inference for each case. In the case of a completely instantiated feature vector, this inference can be completed in $O(N)$ time; otherwise BN inference can be used. Unfortunately for almost all realistic partial orderings M^* will be a highly dense network and memory requirements will be prohibitive. Furthermore, the network structure S^* would be non-interpretable by a human. The intractability of M^* is a central problem with applying this method in practice, and we devote most of the remainder of this paper to presenting ways to remedy this problem.

3.1 Model Averaging over Naïve structures

One popular class of models that fits into the $\mathcal{L}_k^\pi$ schema is the class of naïve (simple) Bayes models. The naïve classifier is a probabilistic model that accomplishes classification by making the assumption that any feature $F_i \in F$ is conditionally independent of any other feature $F_j \in F$ given the value of the class variable C. The naïve model can be represented by the Bayesian network shown in Figure 1.

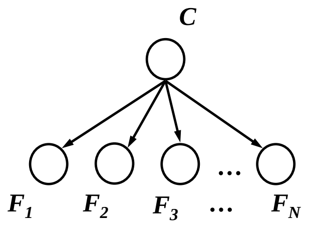

Figure 1: A naïve network: C is the class node which can take on one value for each possible class, and the F_i denote features of interest.

Naïve classifiers have several desirable features: First, they are simple to construct, requiring very little domain background knowledge, as opposed to general Bayesian networks which can require numerous intensive sessions with experts, or a large real-world database to learn the dependence structure between features. Second, naïve networks can be built with very constrained space and time complexity: constructing a network requires the estimation of a set of $O(N \cdot R \cdot N_c)$ parameters, where R is the maximum number of feature states and N_c is the number of class states. Each of which can be estimated from data in time $O(N_D)$, where N_D is the number of records in the database. Inference with naïve networks is also efficient; classification of a new feature vector F' can be performed in time $O(|F'|)$, even if F' is an incomplete instantiation of features.

Despite their simplicity, these classifiers have been shown to perform surprisingly well in practice. Domingos and Pazzani (1997) have shown that naïve classifiers can be optimal (in terms of classifcation accuracy) even when the underlying distribution does not satisfy the naïve assumptions. Friedman (1997) argues that the low variance of the naïve classifier can mitigate the bias, resulting in overall accurate predictions. Finally, Ng and Jordan (2002) show both theoretically and empirically that the naïve classifier converges quickly to its asymptotic error-level. These studies explain why the naïve model has continued to compare favorably to state-of-the-art classification algorithms.

The construction of a naïve classifier given a set F of potential attributes requires only two general steps: (1) Select the subset of features $F' \subseteq F$ judged to be relevant to classification, and (2) Calculate the set $\tilde{\theta}$ of parameters using Equation 4. The feature selection problem (1) is a difficult and central problem in machine learning in general. In terms of naïve classifiers, the selection of the appropriate subset F' has been shown to be both important to classification and non-trivial to perform in practice (Langley and Sage, 1994; Kohavi and John, 1997; Friedman, Geiger, and Goldszmidt, 1997). Obviously eliminating features that do not bear on the classification is important, but also important is the ability to minimize redundant features.

Our method allows us to take a strict Bayesian approach to feature selection; rather than finding a single "good" set F', we can efficiently address the problem of model averaging predictions over all 2^N possible feature-set structures. An enumeration of these different structures is illustrated in Figure 2.

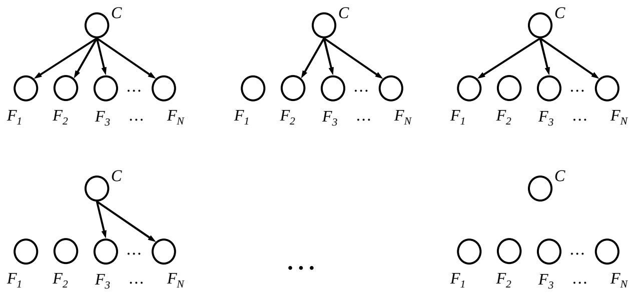

Figure 2: Enumerating all the 2^N possible naïve Bayes net structures.

The summary model M^*, defined in Theorem 3, for the naïve class is especially simple, itself being a naïve network over *all* features. Equation 16 for a naïve Bayes net reduces to

$$\theta^*_{ijk} \propto \tilde{\theta}^{\emptyset}_{ijk} \cdot f_i(\emptyset \mid D) + \tilde{\theta}^{C}_{ijk} \cdot f_i(\{C\} \mid D), \tag{19}$$

where $f_i(\emptyset \mid D)$ and $f_i(\{C\} \mid D)$ are proportional to the local posterior probability of $Pa_i = \emptyset$ and $Pa_i = \{C\}$, respectively. The sufficient statistics and Dirichlet priors required for this calculation are the same as those needed for calculating the parameters of (a) a single network with no arcs present, and (b) a single naïve network with all arcs present. This reparameterization requires order $O(N \cdot N_D)$ time and space requirements, which are the same that are needed to calculate the standard parameters of a single naïve network over all N features. We call a naive structure so parameterized a *Naïve model averaging* (NMA) classifier. In Section 4 we present empirical results showing that this reparameterization, over a wide range of experimental parameters, almost always produced better classification results than a standard naïve model.

3.2 Approximate Model Averaging

As noted in Section 3, a serious practical difficulty with constructing M^* according to Theorem 3 when no ordering is known, is that it requires in the worst case the construction of a completely-connected Bayesian network and inference can thus be intractable for all but small N. An obvious pruning strategy, however, is to truncate the sum in Equation 15 to include no more than n parents. Here we present one possible method for selecting the n most important parents for each node.

If we reorder the possible parent sets for node X_i as $O_P \equiv \{Pa_i^1, \ldots, Pa_i^{\mu_i}\}$ such that $f(X_i, Pa_i^{\nu} \mid D) > f(X_i, Pa_i^{\lambda} \mid D)$ if and only if $\nu < \lambda$, then an approximation for Pa_i^* can be constructed by the following procedure:

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

1. *Let* $Pa_i^* = \emptyset$

2. *For* $\nu = 1$ *to* μ_i,
   if $|Pa_i^* \cup Pa_i^{\nu}| \leq n$, *let* $Pa_i^* = Pa_i^* \cup Pa_i^{\nu}$, *else continue.*

We denote the class of structures being averaged over using this procedure as $\mathcal{L}_{kn}^{\pi}(D)$, and we call the method *Approximate Model Averaging* (AMA). Obviously $\mathcal{L}_{kN}^{\pi}(D) = \mathcal{L}_{k}^{\pi}$. Furthermore, we empirically show in Section 4 that the loss in ROC area, ε, due to this approximation for $n \geq 10$ lies around $-0.6\% \leq \varepsilon \leq 0.6\%$ with 99% confidence for $N \leq 100$ and for a wide range of other parameters. We also show empirically that classifications are not typically sensitive to the value of n, so often n can be made relatively small without degrading classification results.

4. Experimental Tests

In this section we describe several experimental investigations that were designed to test the performance of NMA and AMA on arbitrary distributions. We first generate synthetic data to allow us to more extensively vary parameters, then we perform tests on several real-world machine learning data sets. All experiments were implemented in C++ using code that was based on the *Structural Modelling, Inference and Learning Engine* (SMILE) (Druzdzel, 1999), a library for constructing probabilistic decision support models.[3] Experiments were run on an 1.6 GHz Pentium PC with 1 GB of RAM running Windows XP.

4.1 Experimental Setup

There are at least five parameters for which we sought to characterize the performance of classifier predictions: the number of nodes N, the approximation limit n on the size of the maximum in-degree in the summary network, the maximum in-degree ("density") K of the *generating* network, the maximum in-degree k ($k \leq n$) allowed in models in $\mathcal{L}_{k}^{\pi}$, and the number of records N_D. 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.

In our experiments, four classifiers were compared: AMA using a fixed partial ordering, a NMA classifier, a single naïve network (SNN) with the standard parameterization (Domingos and Pazzani, 1997), and a non-restricted two-stage greedy thick-thin (GTT) model selection over the space of DAGs, which is described below.

The algorithm used to generate the GTT model, which assumes no ordering on the nodes, is as follows:

Procedure 2 (Greedy thick-thin search)
Given: a network S with no arcs.
Do:

1. *Repeatedly add the arc whose addition maximally increases the marginal likelihood $P(D \mid S)$ without creating a cycle until no increase is possible.*

2. *Repeatedly delete the arc whose deletion maximally increases $P(D \mid S)$ until no increase is possible.*

The inner-loop of each test performed the same procedure: Given the six parameters $\{N, N_D, N_{test}, K, n, k\}$, we did the following:

3. SMILE can be downloaded from http://www.sis.pitt.edu/˜genie; however the learning functionality required for our experiments is not yet available for public release.

Procedure 3 (Basic testing loop)
Given: N, N_D, N_{test}, K, n, k.
Do:

1. *Generate a random Bayesian network* $B = G(N,K)$.
2. *Sample* N_D *training records and* N_{test} *test records from the distribution defined by B.*
3. *Train two classifiers to be compared,* M_1 *(typically the AMA classifier) and* M_2 *(the classifier to be compared), on the training records.*
4. *Test* M_1 *and* M_2 *on the test data, measuring the ROC areas* R_1 *and* R_2*, respectively, of each.*
5. *Calculate the quantity* $\delta = \frac{R_1 - R_2}{1 - R_2}$.

The *ROC Area* of a classifier (cf., Egan, 1975), is the area of the curve showing the true-positives of the classifier versus the false-positives as the sensitivity of the classifier is swept out from 0 to 1. It has been used with increasing frequency in machine learning because it provides an objective evaluation of a classifier without requiring the specification of a particular utility function (e.g. zero-one loss).

The performance metric δ indicates what percentage of M_2's missing ROC area $(1 - R_2)$ is covered by M_1: If M_1 is perfect then δ will be 1, if M_1 is equivalent to M_2 then δ will be 0, and if M_1 is worse than M_2 ther

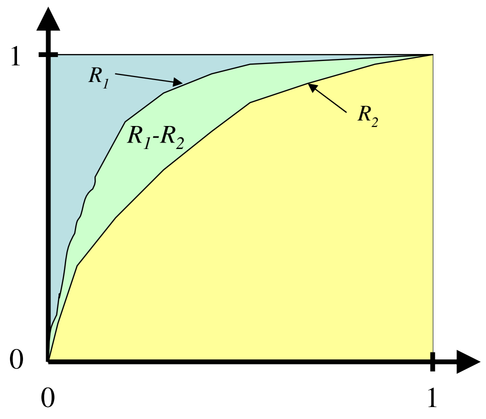

Figure 3: The performance metric δ used in our experiments measures the fraction of ROC area captured by our classifier (with ROC area R_1) versus some other classifier (with ROC area R_2).

parameters, this procedure was repeated N_{trials} times and δ was averaged over these trials.

For some experiments it was necessary to generate networks randomly. 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. The algorithm for selecting network structures at random is as follows:

Procedure 4 (random structure generation)
Given: a set of nodes V; number of records, N_D; and maximum in-degree, K.
Do:

1. *Construct a total ordering $o(V)$ over the variables.*

2. *For each node V_i do:*

   (a) *Choose a number of parents, n_p, uniformly at random from $\{1,\ldots,K\}$.*

   (b) *Select, uniformly at random,* $\min(n_p, o(V_i))$ *parents P such that $o(P) < o(V_i)$.*

The choice of the proper set of noninformative structure priors is non-trivial, and in these experiments we do not attempt to address the subtle complexities inherent in this process. In all cases we assume a uniform prior over non-forbidden structures and thus allow $p_s(X_i, Pa_i) = 1/\mu_i$ for all i. These priors will put overwhelming mass on networks with a "medium" number of arcs because there exist many more of these DAGs. We also adopted the K2 parameter prior which sets $\alpha_{ijk} = 1$ for all (i, j, k). This criterion has the property of weighting all local distributions of parameters uniformly, and has been shown empirically to be an effective non-informative prior (Cooper and Herskovits, 1992; Heckerman et al., 1995). All variables in our synthetic tests were binary, $N_c = R = 2$, and $N_{test} = 1000$, and we typically chose an exponentially increasing sequence of values for N to test the benefits of the algorithms on a wide-range of scenarios. In many experiments we sampled the in-degree of generating graphs K uniformly from the set $\{1, 2, \ldots, N\}$; we use the notation $K \hookleftarrow \{1, \ldots, N\}$ to denote this procedure.

In all experiments, π for AMA 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. In preliminary experiments, these methods produced comparable results, but (2) and (3) performed a few percent better. In all results presented below, method (3) was used to generate π.

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

Symbol	*Description*
N	Number of nodes
N_D	Number of training records
N_{test}	Number of testing records
N_{trials}	Number of times Procedure 3 was repeated
K	Maximum in-degree of generating graphs
k	Maximum in-degree in $\mathcal{L}_k^\pi$
n	Maximum in-degree in summary MA network (Section 3.2)
SNN	Single naïve network (with feature selection)
NMA	Naïve model averaging
GTT	Greedy thick-thin
AMA	Approximate model averaging

Table 1: Table of symbols relevant to experiments.

4.2 Experimental Results

In this section we present a range of experimental tests we performed using data generated from random graphs, data generated from the standard benchmark ALARM network, and data from real-world data sets from the UCI database.

4.2.1 NAÏVE MODEL AVERAGING VERSUS A SINGLE NAIVE NETWORK

It has been shown (Domingos and Pazzani, 1997) that the naïve classifier with the standard parameterization can be optimal under zero-one loss even when the underlying independence assumptions are incorrect. That, together with the fact that calculating parameters for this model can be done with a single pass through the data, make it a useful and widely used model for classification, and it would be interesting if we find a model that has the same ease of calculation but performed significantly better. There are, of course, many other classifiers that we could compare to, in particular, optimizing the conditional likelihood under the naïve network assumptions corresponds to a logistic regression model which has also shown to do well for classification. We defer comparisons of our method to logistic regression and other models for future work.

As Section 3.1 shows, using NMA it is possible to model average over all features sets for a naïve model simply by reparametrizing a single naïve network according to Equation 19. Because of the simplicity of this technique (an existing naïve classifier could trivially be replaced with a model-averaging version just by a change in parameters), we performed an extensive set of comparisons of NMA versus a SNN using synthetic data generated from randomly-constructed (not necessarily naïve) Bayesian networks.

Feature selection for the SNN was performed by successively adding the arc that maximized the posterior probability of the network structure S until no arc resulted in an increase. Although this is a greedy strategy, for SNN under the assumptions taken in this paper, it results in a structure that globally maximizes the posterior probability given the data. This can be seen because, given structure modularity, the marginal likelihood for a given arc is independent of all other arcs in the network.

We tested the relative performance of SNN versus a single naïve model (SNN$_{all}$) over *all* features (i.e., without any feature selection) by performing Procedure 3 with N varied over the set $\{10, 20, 40, 80, 100, 320\}$, with N_D varied over the set $\{100, 200, 400, 800, 1600, 3200, 6400\}$ and with $K \hookleftarrow [1,\ldots,N]$. We found that for all configurations of experimental parameters, SNN dominated SNN$_{all}$, with average $\delta = 29\% \pm 1\%$. Because of this, we do not include SNN$_{all}$ in any of our comparisons to AMA, NMA or GTT.

In the evaluation of NMA performance over SNN performance, we simultaneously varied N from the set $\{10, 20, 40, 80, 160\}$, N_D from the set $\{50, 100, 200, 400, 800, 1600, 3200, 6400\}$, and K from the set $\{5, 10, 20, 40, 80, 160\}$ (obviously however $K \leq N$). We used $N_{trials} = 1000$ in order to establish clear statistical significance. The results are shown in Table 2. The remaining area covered by NMA ranged from a fraction of one percent to 17%, but in all save one of the 160 configurations measured, the improvement of NMA was significant at the 99% level, the general trend being that NMA performed better for smaller values of N and N_D. The lower and upper quartiles (i.e., the 25% and 75% quantiles: the values of δ that confine the middle 50% of the values of δ that we observed in our tests) show the true spread of the data, i.e., because of the large number of trials, the confidence intervals in the mean are not indicative of the widths of the distributions.

4.2.2 MODEL AVERAGING OVER $\mathcal{L}^{\pi}_{kn}(D)$ VERSUS $\mathcal{L}^{\pi}_{k}$

Since, as we have already mentioned, averaging over the full class $\mathcal{L}^{\pi}_{k}$ requires considerable calculation using Equation 14, we must resort in general to using the single summary network and instead only averaging over the class $\mathcal{L}^{\pi}_{kn}(D)$. We were interested in testing what price we pay in terms of classification for reducing the size of the space of models. The first set of experiments we performed tested the degree of error incurred by model averaging over $\mathcal{L}^{\pi}_{kn}(D)$ instead of the full class $\mathcal{L}^{\pi}_{k}$.

$N_D \rightarrow$		50			100			200			400		
N	K	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u
10	5	17.2±1.6	8.4	24.0	17.0±1.4	9.5	22.8	16.5±1.4	8.7	21.7	14.0±1.2	7.3	19.4
10	10	17.1±1.6	7.7	23.6	16.9±1.4	9.2	23.9	16.2±1.4	7.9	23.5	15.5±1.3	7.8	21.2
20	5	8.7±1.1	3.1	12.1	8.7±1.0	3.7	12.2	8.4±0.9	4.1	11.3	9.0±1.0	3.9	13.3
20	10	8.1±1.0	3.2	12.4	10.0±1.3	3.5	13.5	9.6±1.0	4.7	12.5	8.9±1.1	3.3	11.9
20	20	8.5±1.0	3.7	11.9	9.6±1.1	3.8	13.9	10.1±1.1	4.4	13.7	9.8±1.1	4.2	13.4
40	5	3.8±0.8	0.3	7.3	3.5±0.6	0.8	6.2	4.5±0.6	1.7	6.9	4.6±0.6	1.5	5.7
40	10	3.0±0.6	0.2	5.5	4.2±0.6	1.3	6.6	3.8±0.6	0.8	5.7	4.6±0.7	1.5	6.2
40	20	3.0±0.6	0.2	4.8	4.6±0.7	1.5	6.9	4.4±0.9	1.1	6.7	6.1±1.0	2.1	7.9
40	40	2.2±0.5	−0.2	4.2	3.5±0.7	0.6	5.7	4.6±0.7	1.4	6.8	5.9±0.8	2.0	8.1
80	5	2.8±0.7	−0.4	4.9	2.7±0.6	−0.2	5.0	2.4±0.5	0.1	4.3	2.1±0.4	0.3	3.8
80	10	1.6±0.5	−0.6	2.9	1.5±0.5	−0.8	3.7	2.1±0.4	0.0	3.7	2.6±0.4	0.6	4.4
80	20	1.2±0.5	−0.7	2.6	1.3±0.5	−0.7	2.9	1.9±0.4	−0.1	3.6	2.4±0.5	0.2	4.0
80	40	0.7±0.5	−1.1	2.1	1.0±0.4	−1.1	2.8	1.4±0.5	−0.5	3.1	2.3±0.5	0.1	3.9
80	80	0.7±0.5	−1.0	1.9	1.0±0.4	−0.8	2.6	1.8±0.4	−0.1	3.1	2.4±0.5	0.1	4.4
160	5	2.0±0.6	−0.9	3.6	2.2±0.6	−0.9	4.5	2.2±0.5	−0.1	4.1	1.6±0.4	−0.3	3.2
160	10	1.2±0.5	−1.0	2.8	1.8±0.5	−0.5	3.8	1.6±0.4	−0.6	3.8	1.2±0.4	−0.6	2.9
160	20	1.1±0.5	−0.5	2.2	0.9±0.4	−1.0	2.6	1.0±0.4	−0.7	2.9	0.9±0.4	−1.0	2.6
160	40	0.6±0.4	−1.0	1.7	0.5±0.4	−1.3	2.0	0.5±0.4	−1.0	1.9	0.8±0.4	−1.1	2.4
160	80	0.6±0.4	−1.1	1.6	0.5±0.4	−1.3	2.0	0.5±0.3	−1.2	1.9	0.7±0.4	−1.1	2.4
160	160	0.1±0.3	−1.4	1.5	0.7±0.4	−1.1	2.0	0.7±0.3	−0.9	2.3	0.5±0.4	−1.2	2.1

$N_D \rightarrow$		800			1600			3200			6400		
N	K	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u
10	5	13.8±1.3	6.7	19.2	11.5±1.0	5.2	15.8	11.5±1.2	4.3	17.1	10.0±1.3	2.9	14.5
10	10	12.3±1.2	5.5	18.1	10.5±1.2	3.9	14.3	8.4±1.1	2.9	11.3	8.4±1.1	2.4	11.9
20	5	7.7±0.9	2.6	11.7	7.7±1.0	1.8	11.8	5.9±0.9	1.2	8.2	5.7±1.0	0.9	7.8
20	10	8.2±1.0	2.8	12.6	6.0±0.9	1.6	8.5	3.4±0.7	0.7	4.0	3.1±0.6	0.4	3.7
20	20	8.7±1.1	2.8	12.6	7.4±1.0	2.0	11.2	5.8±0.9	1.1	7.5	5.0±0.9	0.6	6.7
40	5	4.6±0.7	1.3	6.3	3.1±0.6	0.7	3.9	3.7±0.7	0.5	4.3	3.4±0.8	0.2	3.6
40	10	4.8±0.7	1.6	6.5	3.5±0.6	1.0	4.4	3.1±0.7	0.5	3.3	2.3±0.6	0.2	2.2
40	20	6.0±0.8	1.8	8.2	6.1±0.9	1.6	8.8	5.4±0.9	0.8	8.0	5.0±1.0	0.5	7.7
40	40	7.5±1.0	2.7	10.0	6.6±1.0	1.7	10.0	7.3±1.0	1.1	11.0	5.9±1.0	0.6	9.6
80	5	2.3±0.4	0.5	3.4	2.0±0.5	0.3	2.7	2.2±0.5	0.2	2.7	1.8±0.5	0.1	1.8
80	10	2.3±0.4	0.6	3.6	2.3±0.4	0.5	3.2	1.9±0.4	0.4	2.4	1.3±0.3	0.1	1.4
80	20	2.7±0.5	0.7	4.5	3.0±0.5	0.8	4.4	3.9±0.6	0.8	5.1	3.6±0.6	0.6	5.1
80	40	2.9±0.5	0.6	4.4	3.9±0.6	0.9	5.6	5.4±0.8	1.3	8.0	5.2±0.7	1.0	7.6
80	80	3.3±0.5	0.9	5.3	4.0±0.6	1.0	6.3	4.9±0.8	1.0	7.5	5.2±0.7	0.7	7.9
160	5	1.3±0.4	−0.1	2.8	0.8±0.3	−0.5	1.9	0.9±0.3	−0.1	1.4	0.9±0.3	0.0	1.2
160	10	1.2±0.3	−0.6	2.7	1.2±0.3	0.0	2.6	0.8±0.3	−0.2	1.7	0.9±0.2	−0.1	1.3
160	20	1.2±0.4	−0.7	3.0	1.2±0.3	−0.4	2.6	1.7±0.4	0.1	2.4	2.1±0.4	0.1	3.3
160	40	1.1±0.3	−0.6	2.7	1.6±0.4	−0.4	2.8	2.6±0.5	0.3	3.9	2.6±0.5	0.3	3.9
160	80	0.7±0.4	−1.2	2.3	1.6±0.4	−0.4	3.2	2.2±0.5	−0.2	4.0	3.0±0.5	0.5	4.7
160	160	1.0±0.4	−0.8	2.5	1.7±0.4	0.0	3.1	2.6±0.5	0.4	4.0	3.4±0.5	0.6	5.5

Table 2: Exploration of NMA performance versus SNN performance as N_D, N and K are varied. The error ranges are 99% confidence intervals in the mean. Q_l and Q_u denote the lower and upper quartiles, respectively.

These experiments tested the sensitivity on the approximation parameter n and the number of nodes N. Setting $N_D = 100$, $N_{trials} = 40$, varying N over the values $\{20, 40, 80, 160\}$, varying n over the values $\{1, 5, 7, 10, 12, 14\}$, varying k over the values $\{1, 2, 3, 4\}$, and $K \hookleftarrow [1, \ldots, N]$. The compiled results are shown in Table 3. The ranges denote the 99% confidence interval of the mean. Table 3 shows that for a wide range of sensible values for these three parameters, we pay little

N	k	n	δ (%)
20	1	7	-0.1 ± 0.4
40	1	7	0.0 ± 0.5
80	1	7	1 ± 1
160	1	7	-2 ± 2
40	1	1	2 ± 2
40	1	5	0.3 ± 0.8
40	1	7	0.0 ± 0.5
40	1	10	0.1 ± 0.6
40	1	12	-0.1 ± 0.4
40	1	14	-0.5 ± 0.3
20	1	7	-0.1 ± 0.4
20	2	7	0.2 ± 0.9
20	3	7	1 ± 1
20	4	7	0 ± 1

Table 3: The values of δ between model averaging over $\mathcal{L}_k^\pi$ and model averaging over $\mathcal{L}_{kn}^\pi(D)$ as various parameters are varied.

classification cost by averaging over $\mathcal{L}_{kn}^\pi(D)$ versus $\mathcal{L}_k^\pi$. Especially interesting is that the difference is small even for quite small values of the approximation level n. Even for $n \gtrsim 5$ the percent remaining area captured by averaging over the full class is less than 1.1% (i.e., $0.3\% + 0.80\%$) with probability $P > 0.99$. Also, 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 $\mathcal{L}_{kn}^\pi(D)$ as k increases. Table 3 shows that over the range of k considered, there is not much sensitivity in the results to k. These results, although not comprehensive in scope, support that averaging over $\mathcal{L}_{kn}^\pi(D)$ does not severely degrade the ROC area relative to model averaging over $\mathcal{L}_k^\pi$. This result is important because only over $\mathcal{L}_{kn}^\pi(D)$ can we select a single tractable model to do model averaging via standard Bayesian network inference.

4.2.3 AMA VERSUS GTT, NMA AND SNN

In terms of classification accuracy, a more practical test of the benefits of AMA is to contrast its performance directly with other algorithms. In our first set of measurements to test this, we generated synthetic data and measured the performance of AMA relative to GTT, NMA and SNN. We varied N over the set $\{20, 40\}$ (much higher was too time-consuming for the complete range of k and n below), and simultaneously varied N_D from $\{100, 1000\}$, k from 2–5, n from 4–14 and $K \hookleftarrow \{1, \ldots, N\}$. In these experiments, N_{trials} varied from 50 to ~ 100, depending on the speed at which the AMA model could be learned. The results for $N = 20$ and 40 are shown in Table 4 and 5, respectively. In these and all subsequent tables (except where explicitly stated), the error ranges represent 99% confidence intervals, and Q_l and Q_u denote the lower and upper quartiles, respectively.

These results illustrate several points. First is that AMA performance is relatively insensitive to the value of n. This is an encouraging result, because an approximation network with a large maximum in-degree can result in slow inference when the feature vector to be classified is not complete and N is large. Next we note the evident complementarity between the naïve classifiers (both NMA and SNN) versus GTT. The naïve classifiers perform consistently better at low N_D; whereas GTT does better at high values of N_D. This effect may be due to the difficulty of reliably extracting structural information from very small databases; in which case GTT is not able to generalize well. A final observation about these tables is that for high N_D, GTT can achieve statistically significant and large gains compared to AMA when k is sufficiently small. This effect is more important as N increases because it becomes more difficult to average over $\mathcal{L}_{kn}^{\pi}(D)$ for a fixed k as N is increased. For example, in our experiments it was too computationally costly for us to repeatedly apply AMA enough to get statistically significant measurements for ($N = 40$, $k = 5$). A similar but weaker effect is seen for the naïve classifiers. As N gets large the benefits of AMA appear to lose statistical significance at low N_D; however rarely did the naïve classifiers outperform AMA in a statistically significant sense.

Figure 4 summarizes the results of Tables 4–5 by showing the qualitative rankings of each algorithm as various parameters are varied. These rankings were derived by examining the results in Tables 4–5 and for each classifier $C_i \in \{\text{NMA, GTT, SNN }\}$, calculating δ_i which is the average over all values of n for the particular configuration being considered in Figure 4. The following rules were applied to determine the rankings:

1. If AMA scored significantly better than C_i for a majority of runs, then AMA is ranked higher than C_i, and visa-versa.

2. If $\bar{\delta}_i < \bar{\delta}_j$ then C_i is ranked above C_j.

It is clear from this figure that the quality of AMA classifications depends strongly on both k and N_D.

While synthetic experiments are attractive because they allow us to systematically vary parameters and generate enough samples to achieve statistical significance, they do not necessary reflect performance in the real world. Also, our synthetic data generation process assumed no hidden variables, a fact which might bias our results. To this end we tested the four classifiers on 34 data sets taken from the UCI online database (Blake and Merz, 1998). These results are shown in Table 6.

Here the score δ_d^i for classifier C_i was calculated according to Procedure 3, 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. The scores in Table 6 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.

Configuration			vs. NMA			vs. GTT			vs. SNN		
N_D	k	n	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u
100	2	4	3 ± 3	-4	7	9 ± 3	0	18	13 ± 3	5	19
100	2	6	6 ± 3	-2	10	9 ± 4	2	16	16 ± 4	5	26
100	2	8	5 ± 3	-2	8	8 ± 3	2	16	14 ± 3	5	20
100	2	10	7 ± 3	-1	13	10 ± 3	2	18	15 ± 4	4	24
100	2	12	5 ± 2	-2	9	11 ± 3	4	18	13 ± 3	6	20
100	2	14	7 ± 3	-1	9	11 ± 4	4	18	16 ± 3	6	24
100	3	4	4 ± 4	-4	10	8 ± 3	2	12	13 ± 4	2	21
100	3	6	6 ± 4	-2	13	12 ± 3	4	17	14 ± 4	6	23
100	3	8	9 ± 4	-2	16	13 ± 3	5	18	19 ± 4	7	27
100	3	10	8 ± 4	-2	17	11 ± 3	3	16	17 ± 4	6	24
100	3	12	7 ± 3	-1	9	10 ± 2	3	14	17 ± 3	6	28
100	3	14	6 ± 4	-3	12	12 ± 3	5	18	15 ± 4	3	22
100	4	4	5 ± 4	-4	14	8 ± 3	2	12	15 ± 4	4	22
100	4	6	5 ± 3	-2	12	9 ± 2	3	15	15 ± 3	6	24
100	4	8	5 ± 4	-2	11	10 ± 3	3	14	13 ± 4	3	23
100	4	10	7 ± 3	-2	13	12 ± 3	3	17	16 ± 3	6	23
100	4	12	4 ± 3	-3	8	9 ± 2	3	14	13 ± 3	4	21
100	4	14	8 ± 4	-1	15	12 ± 3	4	18	17 ± 3	8	27
100	5	6	7 ± 4	-3	16	10 ± 3	3	13	15 ± 4	4	23
100	5	8	5 ± 3	-3	10	10 ± 2	5	13	13 ± 3	3	21
100	5	10	4 ± 4	-2	11	9 ± 3	3	14	12 ± 4	4	20
100	5	12	6 ± 3	-2	10	12 ± 3	4	17	16 ± 3	7	24
100	5	14	6 ± 3	-2	9	10 ± 2	4	15	15 ± 3	6	19
1000	2	4	10 ± 4	-2	18	-8 ± 5	-21	10	17 ± 3	6	26
1000	2	6	11 ± 3	1	19	-12 ± 7	-26	11	19 ± 3	11	27
1000	2	8	10 ± 3	1	17	-15 ± 7	-33	7	17 ± 3	7	24
1000	2	10	12 ± 4	2	23	-8 ± 8	-17	13	20 ± 4	13	30
1000	2	12	10 ± 3	2	18	-10 ± 7	-23	10	16 ± 3	9	23
1000	2	14	10 ± 3	2	18	-7 ± 5	-21	10	18 ± 3	11	23
1000	3	4	14 ± 4	2	23	1 ± 4	-8	9	23 ± 4	11	34
1000	3	6	15 ± 4	2	26	-2 ± 5	-7	9	21 ± 3	11	31
1000	3	8	20 ± 4	6	31	2 ± 4	-6	14	27 ± 4	16	35
1000	3	10	15 ± 3	3	23	0 ± 5	-9	13	21 ± 3	13	28
1000	3	12	18 ± 4	5	29	4 ± 4	-3	14	24 ± 4	13	33
1000	3	14	18 ± 4	4	28	1 ± 4	-3	12	24 ± 3	13	33
1000	4	4	23 ± 5	5	37	6 ± 3	1	12	29 ± 4	15	42
1000	4	6	20 ± 5	2	34	8 ± 3	2	13	29 ± 4	14	41
1000	4	8	24 ± 5	5	40	8 ± 3	2	12	31 ± 4	15	45
1000	4	10	21 ± 4	8	32	6 ± 3	1	15	28 ± 3	18	37
1000	4	12	20 ± 4	8	32	7 ± 2	1	14	27 ± 3	18	37
1000	4	14	25 ± 4	7	37	8 ± 3	1	14	31 ± 4	17	45
1000	5	6	23 ± 4	6	38	9 ± 3	3	15	28 ± 4	16	39
1000	5	8	23 ± 5	6	37	10 ± 3	3	14	29 ± 4	15	42
1000	5	10	26 ± 5	9	40	9 ± 2	3	12	32 ± 4	18	45
1000	5	12	25 ± 4	12	38	10 ± 3	2	16	31 ± 4	20	42
1000	5	14	23 ± 4	8	38	9 ± 3	2	16	28 ± 4	15	42

Table 4: Exploration of AMA performance for $N = 20$ as N_D, k and n are varied. Error ranges denote the 99% confidence intervals; Q_l and Q_u denote the lower and upper quartiles, respectively.

Configuration			vs. NMA			vs. GTT			vs. SNN		
N_D	k	n	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u	δ (%)	Q_l	Q_u
100	2	4	1 ± 3	-5	5	6 ± 4	-1	9	5 ± 3	-2	9
100	2	6	3 ± 3	-2	8	5 ± 4	-2	13	6 ± 4	0	13
100	2	8	3 ± 4	-5	10	7 ± 4	-1	10	6 ± 4	-2	15
100	2	10	3 ± 5	-4	4	7 ± 4	1	10	8 ± 5	1	11
100	3	4	1 ± 5	-7	4	6 ± 3	0	6	5 ± 5	-4	8
100	3	6	1 ± 4	-5	5	5 ± 3	-2	9	5 ± 4	-3	10
100	3	8	1 ± 4	-6	4	4 ± 3	-1	9	4 ± 4	-3	9
100	3	10	3 ± 4	-4	7	8 ± 3	1	15	6 ± 4	1	9
100	4	4	-5 ± 5	-10	1	2 ± 4	-3	8	-1 ± 5	-7	5
100	4	6	0 ± 5	-7	7	4 ± 4	-1	6	3 ± 5	-3	6
100	4	8	0 ± 3	-6	5	5 ± 3	0	11	5 ± 4	-1	9
100	4	10	2 ± 5	-4	3	4 ± 2	0	7	6 ± 5	-2	10
1000	2	4	7 ± 4	0	12	-10 ± 8	-21	6	11 ± 4	3	16
1000	2	6	8 ± 4	-1	12	-8 ± 9	-22	7	14 ± 4	5	19
1000	2	8	11 ± 5	1	19	-10 ± 11	-38	13	17 ± 5	8	27
1000	2	10	5 ± 4	-2	11	-15 ± 14	-29	8	11 ± 5	3	20
1000	3	4	14 ± 5	2	26	-3 ± 6	-9	7	19 ± 5	6	29
1000	3	6	19 ± 6	6	30	0 ± 5	-6	7	23 ± 6	11	32
1000	3	8	18 ± 5	7	27	3 ± 6	-4	14	23 ± 4	12	31
1000	3	10	12 ± 5	1	18	0 ± 7	-1	11	18 ± 4	8	25
1000	4	4	19 ± 6	2	29	10 ± 4	2	17	25 ± 5	12	35
1000	4	6	20 ± 5	6	33	6 ± 4	-1	11	25 ± 5	18	35
1000	4	8	18 ± 6	1	28	9 ± 4	3	12	23 ± 5	9	29
1000	4	10	22 ± 6	1	37	8 ± 4	1	12	28 ± 5	13	42

Table 5: Exploration of AMA performance over parameter space for $N = 40$.

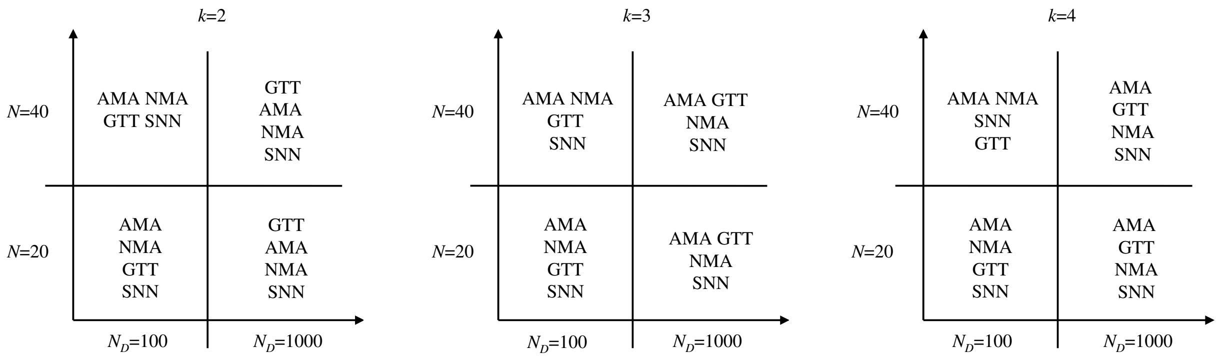

Figure 4: Qualitative comparison showing the ranking of the four algorithms as the number of nodes, the number of records and k are varied.

Data set	δ^{SNN}	δ^{GTT}	δ^{NMA}	δ^{AMA}	N	k	N_D	method
haberman	0.35	0.35	**0.00**	**0.00**	4	4	306	LOO
servo	0.56	0.66	**0.15**	**0.00**	5	5	167	LOO
lenses	0.37	0.45	**0.00**	**0.03**	6	6	24	LOO
hayes-roth	0.32	0.32	**0.00**	**0.01**	6	6	132	LOO
liver-disorders	0.14	0.07	**0.00**	**0.03**	7	7	345	LOO
monks-3	0.83	**0.24**	0.82	**0.00**	7	7	552	T&T
monks-1	0.98	**0.00**	0.98	**0.00**	7	7	554	T&T
monks-2	0.48	0.48	**0.21**	**0.00**	7	7	600	T&T
chess krkopt	0.54	**0.00**	0.54	**0.32**	7	7	28055	CV2
ecoli	0.03	**0.01**	0.02	**0.00**	8	8	336	LOO
yeast	**0.04**	0.11	**0.04**	0.07	8	8	1484	CV2
post-operative	**0.08**	0.46	**0.01**	0.09	9	9	90	LOO
prima-indian diab	**0.01**	**0.01**	**0.01**	0.02	9	9	768	CV2
abalone	0.12	0.08	**0.05**	**0.00**	9	9	4176	CV2
cpu-performance	0.13	0.31	**0.01**	**0.11**	10	10	209	CV2
glass	**0.10**	**0.04**	0.15	0.13	10	10	214	CV2
cmc	**0.01**	0.07	**0.01**	0.04	10	10	1473	CV2
sol-flare-C	0.03	0.09	**0.02**	**0.01**	11	11	322	CV2
sol-flare-M	**0.00**	0.44	**0.17**	0.20	11	11	322	CV2
sol-flare-X	**0.06**	**0.01**	0.18	0.33	11	11	322	CV2
page-blocks	0.30	**0.12**	0.23	**0.02**	11	11	5473	CV2
wine	0.14	**0.01**	0.16	**0.06**	14	7	177	CV2
heart-disease	**0.00**	**0.07**	0.11	0.19	14	6	294	CV2
housing	0.20	**0.06**	0.22	**0.00**	14	5	506	CV2
credit-screening	**0.00**	0.12	0.09	**0.02**	16	5	652	CV2
pendigits	0.58	**0.00**	0.58	**0.00**	17	6	7495	T&T
letter-recognit	0.38	**0.00**	0.38	**0.01**	17	5	20000	T&T
thyroid-disease	0.17	0.28	**0.00**	**0.11**	21	5	7200	T&T
soybean-small	**0.00**	**0.00**	**0.00**	**0.00**	22	4	47	CV4
mushroom	0.86	**0.00**	0.89	**0.03**	22	4	8124	CV2
spect	0.18	0.38	**0.16**	**0.00**	23	4	267	T&T
brst-canc-wisc	0.28	**0.00**	0.29	**0.23**	32	3	569	CV2
connect-4	**0.49**	**0.00**	**0.49**	0.52	43	2	67557	CV2
spambase	0.24	0.25	**0.00**	**0.10**	58	2	4600	CV2

Table 6: Experimental results for 34 UCI data sets. The top scoring classifier for each data set is underlined, the top two are shown in bold. AMA scored in the top one 13 of 34 times compared to 7, 12 and 12 for SNN, GTT and NMA, respectively. It scored in the top two 25 times compared to 11, 17, and 19 for NMA, GTT and SNN, respectively.

The average difference Δ^i between classifier i and AMA: $\Delta^i \equiv \frac{1}{34}\sum_d(\delta_d^{AMA} - \delta_d^i)$, was calculated to gauge the statistical significance of these experiments. The results are shown in Table 7. AMA benefits over the naïve models were significant at the 99% level, but only at the 95% level for GTT.

i	Δ^i (%)	Q_l	Q_u
SNN	19 ± 5	0	33
NMA	13 ± 5	0	23
GTT	8 ± 4	0	20

Table 7: Compiled UCI results. The error ranges denote the error of the mean.

Finally, 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_D systematically varied. The results in Table 8 are shown for classification on the *kinked tube (kt)* and *anaphylaxis (an)* diagnostic nodes. Here, for small number

N_D	δ^{kt} (%)	Q_l^{kt}	Q_u^{kt}	δ^{an} (%)	Q_l^{an}	Q_u^{an}
50	32 ± 13	24	55	3 ± 3	-9	17
100	23 ± 11	9	53	1 ± 3	-11	16
200	13 ± 9	-1	32	-3 ± 4	-17	16
400	12 ± 7	-1	34	-3 ± 5	-21	18
800	4 ± 7	-9	23	2 ± 5	-11	21
3200	0 ± 14	-19	15	6 ± 7	-8	19

Table 8: AMA performance v.s. GTT on synthetic data generated using the ALARM network and classifying on *kinked tube* (kt) and *anaphylaxis* (an).

of records, AMA outperformed GTT at the 99% significance level classifying on the *kinked tube* node; however, it showed no improvement when classifying on the *anaphylaxis* node. These results are notable 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. Precisely what features of the classification node are important is an open question for future research. The prior probability of *anaphylaxis* was about 4 times smaller than that of *kinked tube*; however, the local topology of the network may play a factor as well.

Obviously, using AMA was not without cost. The time to construct the models (and memory requirements) appeared to grow exponentially with k, as shown in Table 9 for $N = 40$.

5. Discussion

We have shown that, under certain assumptions, it is possible to construct a single Bayesian network model, M, whose joint distribution will be identical to exact model averaging over the class, $\mathcal{L}_k^\pi$, of models consistent with a partial ordering π and having in-degree bounded by k. Although for most partial orderings, M will be intractable to build and use for inference, we have demonstrated two ways of putting this technique to practical use: first, by constructing a single network with a particular parameterization that produces approximate model averaged predictions, and second by applying the method to the class of naïve Bayes models, leading to a simple re-parameterization

N_D	Algorithm	train time	test time
100	AMA-4	330	0.101
	AMA-3	27	0.038
	AMA-2	2.6	0.033
	GTT	0.3	0.025
	NMA	0.2	0.021
	SNN	0.02	0.017
1000	AMA-4	1000	0.113
	AMA-3	96	0.035
	AMA-2	8.9	0.033
	GTT	1.2	0.024
	NMA	0.2	0.022
	SNN	0.04	0.017

Table 9: Average training and testing times in seconds for the different algorithms with $N = 40$. “AMA-m" refers to the average over all runs with $k = m$.

that produces predictions equivalent to model averaging over all feature sets, effectively solving the feature selection problem for naïve models in a Bayesian framework.

As an example of the utility of this method, we performed some empirical studies in a classification context, and showed that on both synthetic and UCI datasets, even with relatively little effort in choosing a good value for π and with simple noninformative priors, classifications can be beneficial compared to other common BN classifiers. We have also demonstrated empirically that classifications obtained by model averaging over all naïve features sets is very likely to be beneficial over a single naïve model chosen by selecting the MAP feature set. It can be expected that these results would improve in real-world situations when expert knowledge about realistic node-orderings and structure and parameter priors can be brought to bear.

Our empirical results provide evidence that Bayesian model averaging can improve prediction over model selection, in contrast to the conclusions drawn by Domingos (2000) that Bayesian learning exacerbates the over-fitting problem. First, in our experiments with naïve classifiers, model averaging clearly produced better predictions compared to model selection in terms of the ROC area. Second, in our experiments with approximate model-averaged (AMA) classifiers, we observed the trend that AMA classifications performed successively better as more and more structures were included in the model averaging (i.e., as k was increased). This conflicts with the assertion that model generalization suffers when more models are considered in the averaging process.

In general, the benefits of AMA were not without cost. Construction times for AMA models were higher than other model types, and were observed empirically to grow exponentially as the maximum in-degree k increased. Furthermore, when the approximation parameter n is large, inference with incomplete feature vectors can become prohibitive. The latter observation is mitigated by the fact that the AMA classifier is generally insensitive to the value of n, allowing n to be minimized without sacrificing classification accuracy. However, in cases where the number of nodes, N, is very large, the cost of building the AMA classifier might outweigh the benefits: in this case, if the number of records, N_D, is small then a naïve model-averaged (NMA) classifier would probably be the most attractive since it performed comparably to AMA with orders of magnitude faster training time.

Once the model-averaging model is built, however, the technique has an advantage 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. This is especially relevant in cases where a naïve classifier is currently being employed, as building a NMA classifier retains the same linear time and space complexity required for building a naïve model.

As already stated, when n is large enough the approximation network can be extremely dense, thus making inference difficult when the feature-vector is incomplete. One way to get around this issue, when the incompleteness of the feature-vector is regular, is to learn separate model-averaging models on subsets of data in which the same set of features is missing. Thus a one-time investment of building several feature-vector-specific models would allow us to do $O(N)$ inference even for large n-values.

Future work includes finding a better method for optimizing the ordering π, possibly by doing a search over orderings as in (Friedman and Koller, 2003), and perhaps using cached sufficient statistics with advanced data structures such as ADTrees (Moore and Lee, 1998) to increase the practical limits of k. There are a wealth of other classifiers that it would be interesting to compare with our approach: both non-probabilistic based models such as C4.5, neural networks, support-vector machines, etc., and other model-averaging techniques such as those presented in Madigan and Raftery (1994) and Cerquides (2003)

It should also be possible to relax the assumption of complete training data by using the EM algorithm or MCMC sampling to estimate parameters from data. Finally, the identification of other classes of BN models that easily fit within the $\mathcal{L}_k^\pi$ class could lead to other especially efficient solutions such as that obtained with the naïve Bayes model.

Acknowledgments

We would like to acknowledge the reviewers for their very thorough comments of this paper. This research was supported in part by grant RO1-GM61992-01A1 from the National Institutes of Health and by grant IIS-0325581 from the National Science Foundation.

References

C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. URL `http://www.ics.uci.edu/~mlearn/MLRepository.html`.

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.

Jesús Cerquides. *Improving Bayesian network classifiers*. PhD thesis, Technical University of Catalonia, 2003.

Jesús Cerquides and López de Màntaras. Tractable Bayesian learning of tree augmented naive Bayes classifiers. *Proceedings of the Twentieth International Conference on Machine Learning (ICML 2000)*, pages 75–82, 2003.

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

Denver Dash and Gregory F. Cooper. Exact model averaging with naive Bayesian classifiers. *Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002)*, pages 91–98, January 2002.

Denver Dash and Gregory F. Cooper. Exact model averaging with discrete Bayesian classifiers. In C. M. Bishop and B. J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, Florida, 2003.

Pedro Domingos. Bayesian averaging of classifiers and the overfitting problem. *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000)*, pages 223–230, 2000.

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

Marek J. Druzdzel. SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: A development environment for graphical decision-theoretic models. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI–99)*, pages 902–903, Orlando, FL, July 18–22 1999.

Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, New York, NY, 1973.

J. P. Egan. *Signal Detection Theory and ROC Analysis*. Academic Press, New York, New York, 1975.

Jerome Friedman. On bias, variance, 0/1–loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery*, 1:55–77, 1997.

Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2–3):131–163, 1997.

Nir Friedman and Daphne Koller. Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1–2):95–125, 2003.

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

Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97 (1-2):273–324, 1997. URL `citeseer.nj.nec.com/kohavi96wrappers.html`.

Pat Langley and Stephanie Sage. Induction of selective Bayesian classifiers. In *Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–94)*, pages 399–406, San Francisco, CA, 1994. Morgan Kaufmann Publishers.

Steffen L. Lauritzen and David J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B (Methodological)*, 50(2):157–224, 1988.

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.

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

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.

Andrew Moore and Mary Soon Lee. Cached sufficient statistics for efficient machine learning with large datasets. *Journal of Artificial Intelligence Research*, 8:67–91, March 1998.

Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 841–848, Cambridge, MA, 2002. MIT Press.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1988.

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

T. S. Verma and Judea Pearl. Equivalence and synthesis of causal models. In P. P. Bonissone, M. Henrion, L. N. Kanal, and J. F. Lemmer, editors, *Uncertainty in Artificial Intelligence 6*, pages 255 –269. Elsevier Science Publishing Company, Inc., New York, N. Y., 1991.

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