Boosted Bayesian Network Classifiers

Yushi Jing

College of Computing
Georgia Institute of Technology
Atlanta, GA, 30332

Vladimir Pavlović

Department of Computer Science
Rutgers University
Piscataway, NJ, 08854

James M. Rehg

College of Computing
Georgia Institute of Technology
Atlanta, GA, 30332

Abstract— **The use of Bayesian networks for classification problems has received significant recent attention. Although computationally efficient, the standard maximum likelihood learning method tends to be suboptimal due to the mismatch between its optimization criteria (data likelihood) and the actual goal of classification (label prediction accuracy). Recent approaches to optimizing classification performance during parameter or structure learning show promise, but lack the favorable computational properties of maximum likelihood learning. In this paper we present Boosted Bayesian Network Classifiers, a framework to combine discriminative data-weighting with generative training of intermediate models. We show that Boosted Bayesian network Classifiers encompass the basic generative models in isolation, but improve their classification performance when the model structure is suboptimal. This framework can be easily extended to temporal Bayesian network models including HMM and DBN. On a large suite of benchmark data-sets, this approach outperforms generative graphical models such as naive Bayes, TAN, unrestricted Bayesian network and DBN in classification accuracy. Boosted Bayesian network classifiers have comparable or better performance in comparison to other discriminatively trained graphical models including ELR-NB, ELR-TAN, BNC-2P, BNC-MDL and CRF. Furthermore, boosted Bayesian networks require significantly less training time than all of the competing methods.**

I. Introduction

A Bayesian network is an annotated directed graph that encodes the probabilistic relationships among variables of interest [42]. The explicit representation of probabilistic relations can exploit the structure of the problem domain, making it easier to incorporate domain knowledge into the model design. In addition, the Bayesian network has a modular and intuitive graphical representation which is very beneficial in decomposing a large and complex problem representation into several smaller, self-contained models for tractability and efficiency. Furthermore, the probabilistic representation combines naturally with the EM algorithm to address problems with missing data. These advantages of Bayesian networks and generative models as a whole, make them an attractive modelling choice.

In many problem domains where a Bayesian network is applicable and desirable, we may want to infer the label(s) for a subset of the variables (class variables) given an instantiation of the rest (attributes). Bayesian network classifiers [18] model the conditional distribution of the class variables given the attributes and predict the class with the highest conditional probability. Bayesian network classifiers have been applied successfully in many application areas including computational molecular biology [49] [38] [28], computer vision [51] [44] [48], relational databases [19], text processing [11] [35] [31], audio processing [43] and sensor fusion [40]. Its simplest form, the naive Bayes classifier, has received significant amount of attention [33] [15] [37].

However, standard Maximum Likelihood (ML) parameter learning in Bayesian network classifiers tends to be suboptimal [18]. It optimizes the joint likelihood, rather than the conditional likelihood, a score more closely related to the classification task. Unlike the joint likelihood, however, the conditional likelihood cannot be expressed in a log linear form, therefore no closed form solution is available for the optimal parameters. Recently there has been substantial interest in discriminative training of generative models coupled with advances in discriminative optimization methods for complex graphical models [23] [35] [31] [6] [2] [50].

Under the correct model structure, the parameters that maximize the likelihood also maximize the conditional likelihood (see section III). For this reason, structure learning [10] [25] [20] [7] [26] [32] can potentially be used to improve the classification accuracy. However, experiments show that learning an unrestricted Bayesian network fails to outperform naive Bayes in classification accuracy on a large sample of benchmark data [18] [24]. Friedman et al. attribute this to the mismatch between the structure selection criteria (data likelihood) and the actual goal for classification (label prediction accuracy). They proposed Tree Augmented Naive Bayes (TAN) [18], a structure learning algorithm that learns a maximum spanning tree from the attributes, but retains naive Bayes model as part of its structure to bias towards the estimation of conditional distribution. BNC-2P [24], on the other hand, is a heuristic structure learning method with a discriminative scoring function. Since BNC-2P relaxes the tree structure assumption of TAN and directly maximizes the conditional likelihood, it is shown to outperform naive Bayes, TAN, and generatively trained unrestricted networks. Although the structures in TAN and BNC-2P are selected discriminatively, the parameters are trained via ML training for computational efficiency.

In this work we propose a new framework for discriminative training of Bayesian networks. Similar to a standard boosting approach, we recursively form an ensemble of classifiers. However in contrast to situations where the weak classifiers are trained discriminatively, the “weak classifiers” in our methods are trained generatively to maximize the likelihood of weighted data. Our approach has two benefits. First, ML training of generative models is dramatically more efficient computationally than discriminative training. By combining maximum likelihood training with discriminative weighting of data, we obtain a computationally efficient method for discriminatively training a general Bayesian network. Second, our classifiers are constructed from generative models. This is important in many practical problems where generative models are desired or appropriate.

This work builds on our earlier effort to combine boosting with Dynamic Bayesian network in the application of audio-visual speaker detection [8] [39]. Preliminary results on the BAN algorithm were published in [27].

The paper makes three contributions:

1) We introduce a new discriminative structure learning method, called Boosted Augmented Naive Bayes (BAN) classifier. We demonstrate that BAN is easy to implement and computationally efficient, with classification accuracy superior to naive Bayes, TAN, BNC-2P, BNC-MDL, HGC and comparable to ELR.
2) We interpret a Boosted Bayesian network classifier as a graphical model consisting of a collection of Ensemble Bayesian Network models, and present the first comprehensive empirical evaluation and comparison of Boosted Naive Bayes against competing methods on a large number of standard datasets.
3) We extend Boosted Bayesian network framework to include temporal models such as Dynamic Bayesian Networks (DBNs) and empirically demonstrate that Boosted-DBN outperforms regular DBN in the tasks of sensor fusion and label sequence predictions. Furthermore, we demonstrate that Boosted-DBN has classification accuracy comparable with Conditional Random Fields (CRFs) [31], at the same time, have less computational cost in training.

This paper is divided into 11 sections. Section 1 through 3 review the formal notations of Bayesian networks and parameter learning methodologies. Section 4 introduces AdaBoost as an effective way to improve the classification accuracy of naive Bayes. Section 5 and 6 extend this work to structure learning and proposes the BAN structure learning algorithm. Section 7 extends this work to temporal models by proposing Boosted Dynamic Bayesian Network Classifiers. Section 8 contains the experiments and analysis for BAN structure learning algorithm and Boosted Dynamic Bayesian Network Classifiers. The last three sections contain related works, conclusions and acknowledgements.

II. Bayesian Network Classifier

A Bayesian network B is a directed acyclic graph that encodes a joint probability distribution over a set of random variables $\mathbf{X} = \{X_1, X_2, \ldots, X_N\}$ [42]. It is defined by the pair $B = \{G, \theta\}$. G is the structure of the Bayesian network. θ is the vector of parameters that quantifies the probabilistic model. B represents a joint distribution $P_B(X)$, factored over the structure of the network where

$$P_B(X) = \prod_{i=1}^{N} P_B(X_i|Pa(X_i)) = \prod_{i=1}^{N} \theta_{X_i|Pa(X_i)}.$$

We set $\theta_{x_i|Pa(x_i)}$ equal to $P_B(x_i|Pa(x_i))$ for each possible value of X_i and $Pa(X_i)$[1]. For notational simplicity, we define a one-to-one relationship between the parameter θ and the entries in the local Conditional Probability Table. Given a set of i.i.d. training data $D = \{x^1, x^2, x^3, \ldots, x^M\}$, the goal of learning a Bayesian network B is to find a $\{G, \theta\}$ that accurately models the distribution of the data. The selection of θ is known as parameter learning and the selection of G is known as structure learning.

The goal of a Bayesian network classifier is to correctly predict the label for class $X_c \in \mathbf{X}$ given a vector of attributes $X_a = \mathbf{X} \backslash X_c$. A Bayesian network classifier models the joint distribution $P(X_c, X_a)$ and converts it to conditional distribution $P(X_c|X_a)$. Prediction for X_c can be obtained by applying an estimator such as MAP to the conditional distribution.

III. Parameter Learning

The Maximum Likelihood (ML) method is one of the most commonly used parameter learning techniques. It chooses the parameter values that maximize the Log Likelihood (LL) score, a measure of how well the model represents the data. Given a set of training data D with M samples and a Bayesian Network structure G with N nodes, the LL score is decomposed as:

$$\begin{aligned}\mathrm{LL_G}(\theta|D) &= \sum_{i=1}^{M} \log P_\theta(D^i) = \sum_{i=1}^{M}\sum_{j=1}^{N} \log \theta_{{x_j}^i|Pa(x_j)^i} \qquad (1)\\ &= M \sum_{\substack{x\in X\\ Pa(x)\in Pa(X)}} \widehat{P}_D(x|Pa(x)) \log \theta_{x|Pa(x)}.\end{aligned}$$

$\mathrm{LL}_G(\theta|D)$ is maximized by simply setting each parameter $\theta_{x|Pa(x)}$ to $\widehat{P}_D(x|Pa(x))$, the empirical distribution of the data D. For this reason, ML parameter learning is computationally efficient and very fast in practice.

However, the goal of a classifier is to accurately predict the label given the attributes, a function that is directly tied to the estimation of the conditional likelihood. Instead of maximizing the LL score, we would prefer to maximize the Conditional Log Likelihood (CLL) score. As pointed out in [18], the LL score factors as

$$\mathrm{LL_G}(\theta|D) = \mathrm{CLL_G}(\theta|D) + \sum_{i=1}^{M} \log P_\theta({x_a}^i),$$

where

$$\begin{aligned}\mathrm{CLL_G}(\theta|D) &= \sum_{i=1}^{M} \log P_\theta({x_c}^i|{x_a}^i) &(2)\\ &= M \sum_{\substack{x_a\in X_a\\ x_c\in X_c}} \widehat{P}_D(x_c x_a) \log P_\theta(x_c|x_a). &(3)\end{aligned}$$

[1] We use capital letters to represent random variable(s) and lowercase letters to represent their corresponding instantiations. Subscripts are used as variable indices and superscripts are used to index the training data. $Pa(X_i)$ represents the parent node of X_i and $Pa^j(X_i)$ is the jth instantiation of $Pa(X_i)$ in the training data. In this paper, we assume all of the variables are discrete and fully observed in the training data.

TABLE I

BOOSTED PARAMETER LEARNING ALGORITHM.

```
1) Given a base structure $G$ and the training data $D$, where $M$ is the number of training cases. $D = \{{x_c}^1{x_a}^1, {x_c}^2{x_a}^2, \ldots, {x_c}^M{x_a}^M\}$ and $x_c \in \{-1, 1\}$.
2) Initialize training data weights with $w_i = 1/M, i = 1, 2, \ldots, M$
3) Repeat for $k = 1, 2, \ldots$
    • Given G, $\theta_k$ is learned through ML parameter learning on the weighted data $D_k$.
    • Compute the weighted error, $err_k = E_w[1_{x_c \neq f_{\theta_k}(x_a)}]$, $\beta_k = 0.5 \log \frac{1-err_k}{err_k}$
    • Update weights $w_i = w_i \exp\{-\beta_k {x_c}^i f_{\theta_k}({x_a}^i)\}$ and normalize.
4) Ensemble output: sign $\sum_k \beta_k f_{\theta_k}(x_a)$
```

Given the correct network structure G, parameters that maximizes LL$_G$ also maximizes CLL$_G$. However, in practice the structure may be incorrect and ML learning will not optimize the CLL score, which can result in a suboptimal classification decision. Equation 3 is maximized when

$$P_\theta(x_c|x_a) = \frac{\theta_{x_c} \prod \theta_{x_a|Pa(x_a)}}{\sum\limits_{x_c} \theta_{x_c} \prod \theta_{x_a|Pa(x_a)}} = \widehat{P}_D(x_c|x_a). \tag{4}$$

However, for a generative model such as a Bayesian network, Equation 4 cannot be expressed in log-linear form and has no closed form solution. A direct optimization approach requires computationally expensive numerical techniques. For example, the ELR method of [23] uses gradient descent and line search to directly maximize the CLL score. However, this approach is unattractive in the presence of a large feature space, especially when used in conjunction with structure learning.

IV. BOOSTED PARAMETER LEARNING

A. Ensemble Model

Instead of maximizing the CLL score for a single Bayesian network model, we are going to take the ensemble approach and maximize the classification performance of the ensemble Bayesian network classifier.

Given the class x_c and the attributes x_a, an ensemble model has the general form:

$$F_{x_c}(x_a) = \sum_{k=1}^{K} \beta_k f_{k,x_c}(x_a). \tag{5}$$

where $f_{k,x_c}(x_a)$ is the classifier confidence on selecting label x_c given x_a, and β_k is its corresponding weight. In the case where $x_c \in \{-1, 1\}$, $f_{k,x_c}(x_a)$ is typically defined as the following:

$$f_{k,x_c}(x_a) = x_c f_k(x_a) \tag{6}$$

where $f_k(x_a)$ is the output of each classifier given x_a. Equation 5 can be expressed as a conditional probability distribution over X_c given the additive model F:

$$P_F(x_c|x_a) = \frac{\exp\{F_{x_c}(x_a)\}}{\sum_{x_c' \in X_c} \exp\{F_{x_c'}(x_a)\}}. \tag{7}$$

In binary classification, Equation 7 is then updated as:

$$\begin{aligned} P_F(x_c|x_a) &= \frac{\exp\{x_c F(x_a)\}}{\exp\{F(x_a)\} + \exp\{-F(x_a)\}} \\ &= \frac{\exp\{x_c F(x_a)\}}{\exp\{x_c F(x_a)\} + \exp\{-x_c F(x_a)\}} \\ &= \frac{1}{1 + \exp\{-2x_c F(x_a)\}}. \end{aligned} \tag{8}$$

Similar to Equation 3, the negative CLL score for the ensemble Bayesian network classifier can be defined as:

$$-\text{CLL}_F(F|D) = \sum_{i=1}^{M} \log \frac{1}{P_F({x_c}^i|{x_a}^i)} \tag{9}$$

$$= M \sum_{\substack{x_a \in X_a \\ x_c \in X_c}} \widehat{P}_D(x_c x_a) \log \frac{1}{P_F(x_c|x_a)} \tag{10}$$

B. Exponential Loss Function as an upper bound on the negative CLL score

As an alternative to the CLL score, we are proposing to minimize the classification error for binary ensemble classifier via the following loss function.

$$Loss_F = \sum_{i=1}^{M} \Theta(-{x_c}^i F({x_a}^i)), \Theta(z) = \begin{cases} 0 & \text{for } z < 0 \\ 1 & \text{otherwise} \end{cases}$$

$Loss_F$ is simply the number of incorrectly predicted class labels in the training data. An upper bound on Equation 11 is given by the following exponential loss function [17]:

$$ELF_F = \sum_{i=1}^{M} \exp\{-{x_c}^i F({x_a}^i)\} \tag{11}$$

Solving for $x_c F(x_a)$ in Equation 8 and combining with Equation 11, we have

$$\text{ELF}_F = \sum_{i=1}^{M} \exp\left\{\frac{1}{2} \log \frac{1 - P_F({x_c}^i|{x_a}^i)}{P_F({x_c}^i|{x_a}^i)}\right\} \tag{12}$$

$$= \sum_{i=1}^{M} \sqrt{\frac{1 - P_F({x_c}^i|{x_a}^i)}{P_F({x_c}^i|{x_a}^i)}}$$

$$= M \sum_{\substack{x_a \in X_a \\ x_c \in X_c}} \widehat{P}_D(x_c x_a) \sqrt{\frac{1}{P_F(x_c|x_a)} - 1} \tag{13}$$

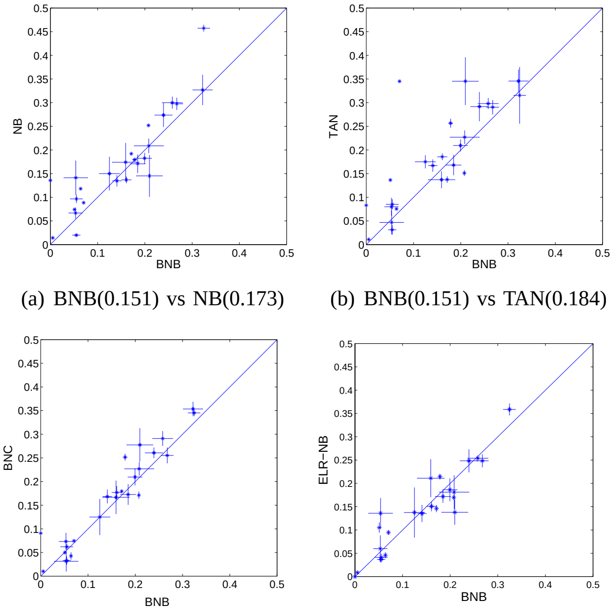
Equation 13 simply leads to a loss function that uses the square root of the inverse conditional distribution of the true training sequence, which can be readily proven as an upper bound for negative CLL score in Equation 10.

C. Boosted Parameter Learning

An ensemble Bayesian network classifier takes the form $F_{\theta,\beta}$ where θ is a collection of parameters in the Bayesian network model and β is the vector of hypothesis weights. We want to minimize ELF$_{\theta,\beta}$ of the ensemble Bayesian network classifier as an alternative way to maximize the CLL score. We used Discrete AdaBoost algorithm, which is proven to greedily and approximately minimize the exponential loss function in Equation 13 [17].

At each iteration of boosting, the weighted data uniquely determines the parameters for each Bayesian network classifier θ_k and the hypothesis weights β_k via efficient ML parameter learning. The algorithm is shown in Table I.

There is no guarantee that AdaBoost will find the global minimum of the ELF. Also, AdaBoost has been shown to be susceptible to label noise [13] [3]. In spite of these issues, boosted classifiers tend to produce excellent results in practice [47] [14]. Boosted Naive Bayes (BNB) has been previously shown to improve the classification accuracy of naive Bayes [16] [45]. In the next section, we demonstrate that BNB outperforms naive Bayes and TAN on a large set of benchmark data.



(a) BNB(0.151) vs NB(0.173) (b) BNB(0.151) vs TAN(0.184)

(c) BNB(0.151) vs BNC-2P(0.164) (d) BNB(0.151) vs ELR-NB(0.161)

Fig. 1. Scatter plots for experiments on 25 sets of UCI and artificial benchmark data. The average testing error is shown next to the method name.

D. Experiments

We evaluated the performance of BNB on 23 datasets from the UCI repository [5] and two artificial data sets, Corral and Mofn, designed by John and Kohavi [30]. Friedman et al., Greiner et al. and later Grossman et al. used this group of data sets as benchmarks for Bayesian network classifiers. We used hold-out test for larger data sets and 5 fold cross validation for smaller sets. Our implementation is based on the BNT toolkit by Kevin Murphy [36]. For binary classification, we used Discrete AdaBoost for parameter boosting. In the multi-class case, we used AdaBoost.MH [17]. The competing Bayesian network classifiers are described below:

- **NB:** naive Bayes.
- **TAN:** Tree Augmented naive Bayes [18].
- **BNC-2P:** Discriminative structure selection via CLL score. [24].
- **ELR-NB:** NB with parameters optimized for conditional log likelihood [23] via gradient descent.

Table IV in Page 9 lists the average testing error for BNB and other Bayesian network classifiers including our novel algorithm BAN, which is introduced in Section 5. Figure 1(a) to 1(d) presents the average testing errors and their corresponding one-standard-deviation bars for competing Bayesian network classifiers. In Figure 1, points above the line $y = x$ correspond to data sets for which BNB outperforms the competing algorithm. The average testing error is shown next to the method name. We applied pairwise t-test on the 25 pairs of average testing errors for competing algorithms to obtain confidence scores.

Figure 1(a) and 1(b) show that BNB has lower average testing error than NB $(p < 0.02)$ and TAN $(p < 0.02)$. Also, we find BNB to slightly outperform the BNC-2P discriminative structure learning algorithm on average testing error $(p < 0.04)$.

We also compared BNB to ELR-NB, a naive Bayes trained using ELR algorithm. The performance scores for ELR-NB were taken from the auxiliary material published online with [23]. From the graph, it is reasonable to conclude that BNB is comparable with ELR-NB on this set of benchmark data. However, BNB has computational complexity asymptotically equivalent to naive Bayes, making it an efficient and simple alternative to ELR-NB.

E. Computational Complexity of BNB

Given a naive Bayesian network with N attributes and training data with M samples, the ML training complexity is $O(NM)$, optimal when every attribute is observed and used for classification. Parameter boosting for a naive Bayes takes $O(NMT)$ where T is the number of iterations of boosting. In our experiments, boosting seems to give good performance with a constant number (10-30) of iterations regardless of the number of attributes. Therefore, the training complexity for BNB is essentially $O(NM)$. This is consistent with the finding of Elkan [16].

In comparison to other discriminative trained Bayesian networks, BNB has the least asymptotic training complexity. For example, TAN, widely regarded as the discriminative structural extension to naive Bayes, has training complexity of $O(N^2M)$.

F. Ensemble Bayesian network classifier

The simplest form of Boosted Bayesian network is the BNB model. As shown in Figure 2, it can be represented as a graphical model. We define a set of hidden binary nodes

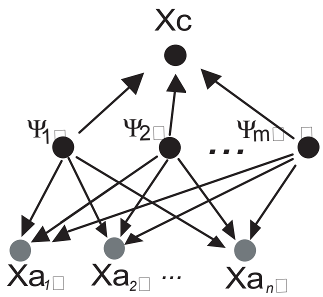


Fig. 2. Graphical representation for Boosted Naive Bayes.

$\psi_i \in \{-1, 1\}$ which correspond to the outputs of the naive Bayes classifier after each iteration of boosting. The lower layer of the graphical model is a set of Bayesian network classifiers, with parameters trained using ML learning on reweighted training data. The top layer is a discriminative model.

From Equation 8, the top layer encodes the conditional distribution for X_c given the value of the hidden nodes ψ_i where

$$P(x_c|\psi_1, \ldots, \psi_N) = \frac{1}{1 + \exp\{-2\sum_k \beta_k \psi_k\}}.$$

.

Since the lower layer model can be any generative model including naive Bayes, TAN, HMM and etc, we call this graphical representation as Ensemble Bayesian Network Classifiers.

Given the excellent performance of BNB, it is natural to ask whether it could be combined with structure learning to further improve the classification performance. In the next section, we introduce BAN, a novel discriminative structure learning algorithm.

V. Structure Learning

Given training data D, structure learning is the task of finding a set of directed edges G that best models the true density of data. In order to avoid overfitting, Bayesian Scoring Function [10] [26] and Minimal Description Length (MDL) [32] are commonly used to evaluate structure candidates. The MDL score is asymptotically equivalent to the Bayesian scoring function in the large sample case and this paper will concentrate on the MDL score. MDL score is defined as

$$MDL(B|D) = \frac{\log |D|}{2}|B| - LL(B|D) \tag{14}$$

where $|B|$ is the total number of parameters in model B, and $|D|$ is the total number of training samples.

Grossman et al. [24] proposed the CMDL scoring function by substituting LL score with CLL score in the second term of Equation 14.

$$CMDL(B|D) = \frac{\log |D|}{2}|B| - CLL(B|D)$$

An exhaustive search over all structures against an evaluation function can in principle find the best Bayesian network model, but in practice, since the structure space is super exponential in the number of variables in the graph, it is not feasible in nontrivial networks. Several tractable heuristic approaches have been proposed to limit the search space. The K-2 [10] algorithm and the variant MCMC-K2 [20] define a node ordering such that a directed edge can only be added from a high ranking node to a low ranking node. Heckerman [26] proposed a hill-climbing local search algorithm to incrementally add, remove or reverse an edge until a local optimum is reached.

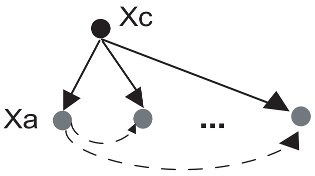


Fig. 3. An example of ANC, the dotted edges are structural extensions to Naive Bayes.

An alternative structure penalty is to simply limit the number of parents an attribute can have. An Augmented Network Classifier (ANC) [29], in which each attribute is limited to have at most one more parent besides the class node, is an example of this approach. Friedman et al. [18], based on the work by [9], proposed an efficient algorithm to construct an optimal Tree Augmented Naive Bayes (TAN), a special case of the ANC model.

K-2, Heckerman's method, ANC and TAN all utilize standard ML parameter learning for simplicity and efficiency.

VI. Boosted Augmented Naive Bayes

Although the training complexity of parameter boosting is within a constant factor of ML learning, combining parameter boosting with structure search is still impractical. Even with constrained search space, hill-climbing search and K-2 algorithm could still search through a large number of structures.

On the other hand, TAN supports efficient learning by limiting the number of parents per attribute to two. TAN augments a standard naive Bayes classifier by adding up to $N - 1$ additional edges between attributes. The additional edges are constructed from a maximal weighted spanning tree with attributes as vertices. The weights are defined as the conditional mutual information $I_p(X_{a_i}; X_{a_j}|X_c)$ between two attributes X_{a_i}, X_{a_j} given the class node X_c where

$$\begin{aligned} & I_p(X_{a_i}; X_{a_j}|X_c) \\ = & \sum_{\substack{x_c \in X_c \\ x_{a_i} \in X_{a_i} x_{a_j} \in X_{a_j}}} P(x_{a_i} x_{a_j} x_c) log \frac{P(x_{a_i} x_{a_j}|x_c)}{P(x_{a_i}|x_c) P(x_{a_j}|x_c)} \end{aligned}$$

TAN learning algorithm constructs the optimal tree-augmented network B_T that maximizes LL$(B_T|D)$. However, the TAN model adds a fixed number of edges regardless of the distribution of the training data. If we can find a simpler model to describe the underlying conditional distribution, then there is usually less chance of over-fitting.

Our BAN learning algorithm extends the TAN approach using parameter boosting. Starting from a naive Bayes model,

TABLE II
BOOSTED AUGMENTED NAIVE BAYES.

1) Given training data D, construct a complete graph G_{full} with attributes X_a as vertices. Calculate $I_p(X_{a_i}; X_{a_j}|X_c)$ for each pair of attributes X_a, $i \neq j$, where

$$I_p(X_{a_i}; X_{a_j}|X_c) = \sum_{\substack{x_{a_i} \in X_{a_i} \\ , x_{a_j} \in X_{a_j}, x_c \in X_c}} P(x_{a_i} x_{a_j} x_c) log \frac{P(x_{a_i} x_{a_j}|x_c)}{P(x_{a_i}|x_c) P(x_{a_j}|x_c)} \tag{15}$$

2) Construct G_{TAN} from G_{full}, set $G_{BAN} =$ naive Bayes, CLL$_{best} = -\inf$.
3) For $k = 1$ to $N-1$
   - Parameter boosting using G_{BAN} as base structure.
   - Evaluate the CLL score for the current G_{BAN}, terminate if the new CLL score is less than CLL$_{best}$.
   - else, update CLL$_{best}$. Remove the edge $\{X_{a_i} X_{a_j}\}$ containing the largest conditional mutual information $I_p(X_{a_i}; X_{a_j}|X_c)$ from G_{TAN} and add it to G_{BAN}.

at iteration k, BAN greedily augments the naive Bayes with k edges with the highest conditional mutual information. We call the resulting structure BAN^k. We then minimize the ELF score of BAN^k classifier with parameter boosting. BAN terminates when the added edge does not improve the CLL score. Since TAN contains $N-1$ augmenting edges, BAN in worst case evaluates $N-1$ structures. This is linear comparing to polynomial number of structures examined by K-2 or Heckerman search. Moreover, in practice, BAN usually terminates after adding 2 to 5 edges into naive Bayes. Therefore, this approach is very efficient.

The algorithm is shown in Table II. Step 1 in BAN algorithm has computational complexity of $O(N^2M)$, where N is the number of attributes and M is the amount of training data. Since we only add a maximum of $N-1$ edges into the network, step 2-4 has worst case complexity of $O(N^2M)$. Thus BAN has $O(N^2M)$ complexity.

The BAN learning algorithm searches and evaluates only a small number of structures, much less than competing algorithms like BNC-2P and BNC-MDL. As a result, the base Bayesian network structure constructed from BAN usually contains fewer edges than other competing structure learning algorithms.

VII. DYNAMIC BAYESIAN NETWORKS.

In this section, we extend the Boosting framework to Dynamic Bayesian Networks (DBNs). We demonstrate that the resulting classifiers outperform generatively trained DBNs in label sequence prediction. We also show that Boosted Dynamic Bayesian networks has classification accuracy comparable with Conditional Random Fields (CRFs) [35], while has less training computational cost.

A. Hidden Markov Models

A Dynamic Bayesian network extends a static Bayesian network by explicitly representing the temporal relationship among the variables. Hidden Markov Models (HMMs) are one of the most commonly used DBN models, with successful application to speech recognition [43], text classification [11] and computational biology [28]. A HMM model contains a state and an attribute variable. We use $\mathbf{X}_c$ to denote the state sequence and $\mathbf{X}_a$ to denote the discrete attribute sequence[2].

HMMs can be decomposed into two components: the attribute model in the form of a static Bayesian network, and the state model that defines the state transition probability. In HMMs, the state variables are distributed according to a Markov process.

DBNs [22] generalizes HMMs by providing a more flexible representation of the dependencies within a time slice and between time slices. DBNs have successful applications to system monitoring [34], gene discovery [53] and computer vision [41]. Structure learning [21] has been discussed for DBN as well.

B. Label Sequence Prediction and Boosted Dynamic Bayesian network

Label sequence prediction is the task of inferring the labels of the state sequence $\mathbf{X}_c$ given an instantiation of the attribute sequence $\mathbf{X}_a$. Typically HMM selects the label by applying an estimator such as MAP to the estimated posterior distribution. As in the static Bayesian network case, ML parameter learning in HMMs is usually suboptimal for classification tasks. Therefore, we extend the boosted parameter learning to Dynamic Bayesian network to form an Ensemble Dynamic Bayesian network classifier.

We propose to minimize the following label sequence prediction loss function:

$$\text{Loss}_F = \sum_{j=1}^{J} \sum_{i=1}^{|D_j|} \Theta(-x_{c_j}^{\,i} F(x_{aj})) \tag{16}$$

where

$$\Theta(z) = \begin{cases} 0 & \text{for } z < 0 \\ 1 & \text{otherwise} \end{cases}$$

Similar to the case of static Bayesian network, Equation 16 can be bounded by

$$\text{ELS}_F = \sum_{j}^{J} \sum_{i=1}^{|D_j|} \exp(-x_{c_j}^{\,i} F(x_{aj})) \tag{17}$$

[2] $\mathbf{X}_a = \{X_a{}^1, X_a{}^2, ..., X_a{}^T\}$ and $\mathbf{X}_c = \{X_c{}^1, X_c{}^2, ..., X_c{}^T\}$, where T is the length of the sequence.

```
1) Given base DBN structure G and $J$ training sequence $D_{1,2,\dots,J}$, where $D_j = \{x_{a}{}_j^1, x_{c}{}_j^1, x_{a}{}_j^2, x_{c}{}_j^2, \dots, x_{a}{}_j^{|D_j|}, x_{c}{}_j^{|D_j|}\}$.
2) Initialize data weights $W$ with uniform distribution across all samples, $w_j^i = 1/N, i = 1, 2, \dots, |D_j|$ where $N = \sum_{i=1}^{J} |D_i|$.
3) Repeat for $k = 1, 2, \dots, K$
    a) $\theta_k$ is learnt through maximum likelihood parameter learning on the weighted data $D_w$.
    b) Compute the combined weighted label error for the training sequences, $err^k = E_w[1_{x_c \neq f_{\theta_k}(x_a)}] = \sum_{i,j} w_j^i \left[ 1_{x_{c}{}_j^i \neq f_{\theta_k}(x_{a}{}_j)} \right]$
    c) $\beta_k = 0.5 log \frac{1-err_k}{err_k}$
    d) Update weights $w_j^i = w_j^i \exp\{\beta_k x_{c}{}_j^i f_{\theta_k}(x_{a}{}_j)\}$ and normalize.
4) Ensemble output: sign $\sum_{k=1}^{K} \beta_k f_{\theta_k}(x_a)$
```

TABLE III

BOOST-DBN TRAINS AN ENSEMBLE DYNAMIC BAYESIAN NETWORK CLASSIFIER TO IMPROVE LABEL PREDICTION ACCURACY.

Given the weak classifier $f_k(x_a)$ at each iteration of boosting, we can see, with easy modification to the proof for Result 1 in [17], that Discrete AdaBoost (Population version) greedily and approximately maximizes Equation 17 by setting the hypothesis weights β as:

$$\beta_k = 0.5 \log \frac{1 - err_k}{err_k}$$

where

$$err_k = E_w[1_{x_c \neq f_{\theta_k}(x_a)}] = \sum_{i,j} w_j^i \left[1_{x_{c}{}_j^i \neq f_{\theta_k}(x_{a}{}_j)} \right].$$

The Boosted DBN parameter learning algorithm is shown in Table III.

Step 3(a) in Table III has computational complexity of $O(NM + C^2M)$, where C is the cardinality of the state space. This is essentially optimal when every feature is used for classification in HMM. Step 3(b) evaluates the given DBN via Forward-Backward algorithm with computational complexity of $O(C^2NM)$. The computational complexity for Boost-DBN parameter learning algorithm is therefore $O(C^2NMT)$, where T is the number of boosting iteration. In our experiments, Boosted DBN parameter learning algorithm converges after 25-30 iterations of boosting.

VIII. EXPERIMENTS

The experiments section is organized into three subsections. Subsection A through C contain experiments and analysis of BAN structure learning algorithm. Subsection D contains experiments and analysis for Boost-DBN algorithm.

A. Experiments on BAN with simulated datasets

We show that when the structure is incorrect, BNB and BAN algorithm can significantly outperform their generative counterparts. We generated a collection of data from binary chain-structured Bayesian network where the parent of each variable X_i is its predecessor X_{i-1}. The class node X_1 is the root of the chain. The chain-structured Bayesian network is shown in Figure 4. We varied the number of attributes and their parameter values to generate 25 datasets with different distributions. Since the attributes are correlated, naive Bayes can sometimes give a suboptimal classification boundary.

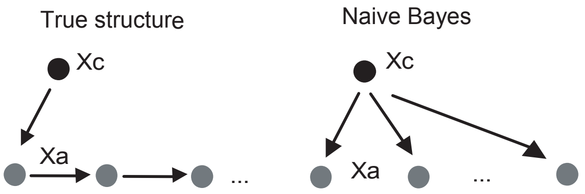


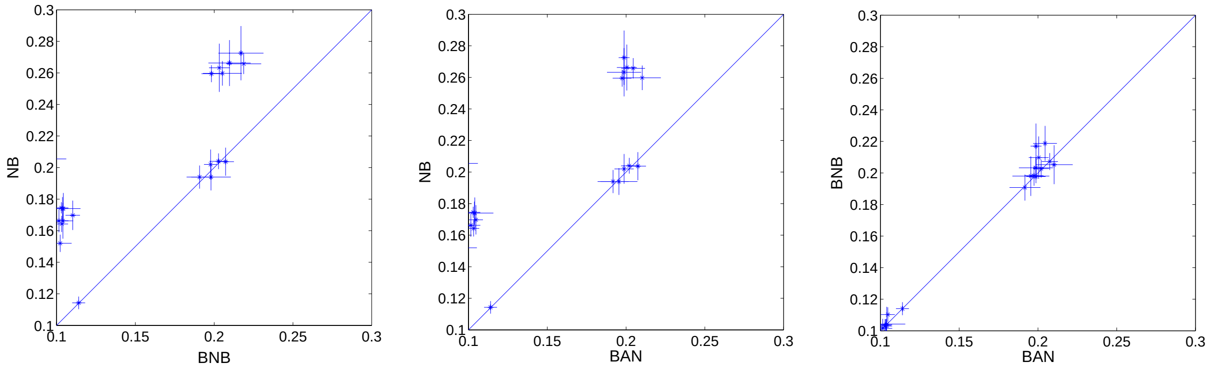
Fig. 4. Data is sampled from chain-structured Bayesian network. Therefore Naive Bayes is a sub-optimal classifier for this dataset.

We present the average testing errors with their one-standard-deviation bar in Figure 6. Figure 6(a) and 6(b) show that BNB and BAN has lower average testing error than NB $(p < 0.005)$. Figure 5 shows the decrease in negative CLL score and testing error in each iteration of parameter boosting. In this dataset, BNB achieves the optimal Bayes error after 8 iterations but the negative CLL score continues to decrease. We want to point out that in 6 out of the 25 datasets, the suboptimal posterior estimation by naive Bayes did not result in label prediction error. In those datasets, NB, BNB and BAN have similar testing error.

As shown in Figure 6(c), the average testing error for BNB is only slightly higher than that of BAN. This is largely because BNB achieved optimal Bayes error in 20 out of the 25 datasets due to the simplicity of our true model. BAN has comparable testing accuracy with BNB in those 20 datasets and has lower average testing error (difference of 2%) than BNB in the remaining 5 datasets. Next section will show that in real-world datasets, where attributes often have complex and strong dependence relationship, BAN outperforms BNB by exploring the structures in the problem domain.

B. Experiments on BAN with UCI datasets

We used the same UCI datasets and evaluation procedures as in Section 4.D to compare the accuracy of BAN with competing algorithms. For our experiments, we implemented BAN, BNB, BNC-2P and TAN, and we used the performance results for BNC-MDL, ELR, C4.5 and HGC from [24] [23]. HGC [25] is a generatively trained unrestricted Bayesian network. ELR-NB and ELR-TAN are Bayesian network



(a) BNB (0.14) vs NB (0.1864)

(b) BAN (0.14) vs NB (0.1864)

(c) BAN (0.14) vs BNB (0.143)

Fig. 6. Test error on simulated experiment. We varied the number of nodes in the chain-structure Bayesian network and their parameter values to generate different distributions (25 sets). Each point in the graph represents the classification accuracy for one particular model distribution. BNB and BAN outperforms NB in 19 out of the 25 simulated datasets. In the remaining 6 datasets, the suboptimal posterior estimation by naive Bayes did not result in label prediction error.

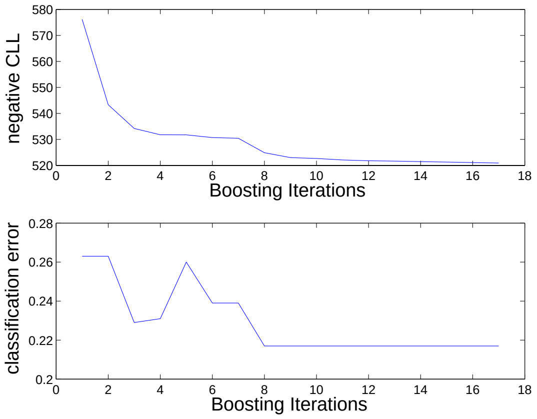


Fig. 5. Negative CLL score and classification error decreases with more boosting iterations.

learned using the ELR algorithm. The scatter plots are shown in Figure 7 and the average testing error is shown in Table IV. All points above $y = x$ are data sets in which BAN outperforms its competitors. The abbreviations for competing algorithms are described below:

- **BAN:** Boosted Augmented Naive Bayes.
- **NB:** naive Bayes.
- **TAN:** Tree Augmented naive Bayes.
- **BNC-2P:** Discriminative structure selection via CLL score. [24].
- **BNC-MDL:** Discriminative structure selection via CMDL score. [24]
- **ELR-NB, ELR-TAN:** NB and TAN with parameters optimized for conditional log likelihood as in Greiner and Zhou [23].
- **HGC:** Generative structure search algorithm from Heckerman et al. [26].

Figure 7(a) and 7(b) show that the average testing error for BAN algorithm is significantly lower than naive Bayes $(p < 0.01)$ and TAN $(p < 0.01)$. BAN also outperforms BNC-2P $(p < 0.005)$ in Figure 7(d). We did not have access to variance data for BNC-MDL, HGC and C4.5. However, since BNC-2P has been previous shown to outperform HGC and BNC-MDL, it seems reasonable to conclude that BAN is superior to HGC and BNC-MDL as well.

As shown in Figure 7(c) and Table IV, BAN has comparable classification accuracy as ELR-NB. However, BAN is much more efficient to train in comparison to ELR-NB and ELR-TAN.

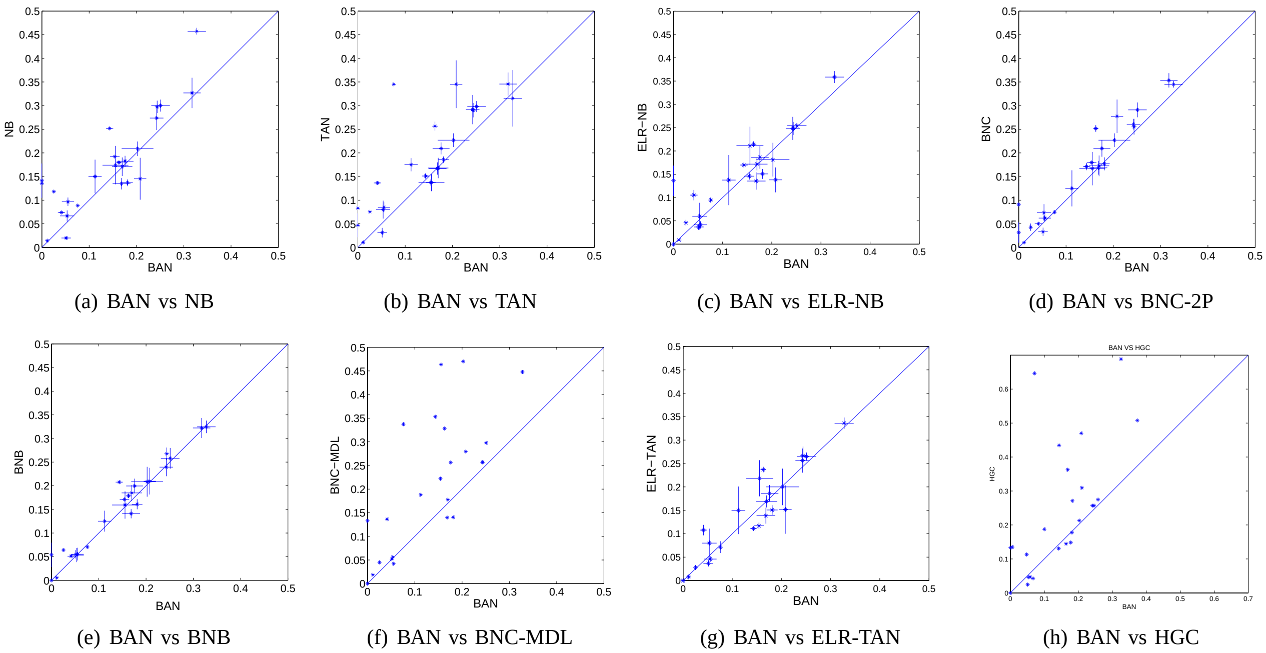
As shown in Figure 7(e), the average testing errors for BAN and BNB are 0.141 and 0.151 respectively. This difference is significant with confidence $p < 0.029$. BAN has lower average testing error (difference of 0.5% - 5%) than BNB in 16 out of the 25 datasets. BNB is better in 6 (difference of 0.5% - 2%) and they tie in 3. Since BAN generalizes BNB, in several datasets (MOFN, IRIS), the structure chosen by BAN is very similar to BNB (with 0 and 1 augmented edges). BAN is more beneficial in datasets where the conditional dependencies among attributes are strong and complex (CORRAL).

This is an interesting result since it shows that combining discriminative structure learning with parameter optimization seems to improve classification accuracy.

C. Discussion

The above experiments demonstrated that boosted parameter optimization in conjunction with greedy structure optimization can improve the classification performance. It is interesting to note that unlike the experimental results in combining ELR with structure learning [24], we find significant benefit in combining parameter boosting with structure learning.

We attribute the success of our approach to the following reasons. First, BAN takes advantage of AdaBoost's resistance to over-fitting [46] and the variance reduction and bias reduction property of ensemble classifiers [52]. Also, as a result of the parameter boosting, the base Bayesian network classifier constructed by BAN is simpler than BNC-2P and TAN. In our experiments, BAN adds 0 to 4 edges to the naive Bayes while BNC-2P typically adds 4 to 16 edges. If both Bayesian networks model the underlying conditional distribution equally well, a simpler structure is usually preferred over a more densely connected one.



(a) BAN vs NB

(b) BAN vs TAN

(c) BAN vs ELR-NB

(d) BAN vs BNC-2P

(e) BAN vs BNB

(f) BAN vs BNC-MDL

(g) BAN vs ELR-TAN

(h) BAN vs HGC

Fig. 7. Scatter plots for experiments on 25 benchmark UCI and artificial datasets.

TABLE IV
TESTING ERROR FOR 25 UCI DATASETS

Name	**BAN**	**BNB**	**TAN**	**NB**	**BNC-2P**	**BNC-MDL**	**NB-ELR**	**TAN-ELR**	**C4.5**	**HGC**
australian	.1812	.1609	.1855	.1368	.1768	.1405	.1488	.1723	.1510	.1445
breast	.0513	.0549	.0312	.0200	.0330	.0518	.0339	.0351	.0610	.0245
chess	.0253	.0640	.0753	.1180	.0428	.0450	.0600	.0375	.0050	.0469
cleve	.1758	.1995	.2095	.1825	.2095	.2563	.1660	.0375	.2060	.2129
corral	.0000	.0538	.0468	.1412	.0314	.0000	.1273	.0771	.0150	.0000
crx	.1684	.1408	.1669	.1347	.1684	.1397	.1505	.1603	.1390	.1308
diabetes	.2438	.2675	.2903	.2974	.2553	.2569	.2419	.2384	.2590	.2569
flare	.1698	.1848	.1679	.1707	.1726	.1776	.1803	.1780	.1730	.1776
german	.2510	.2580	.2980	.3000	.2910	.2977	.2456	.2409	.2710	.2748
glass	.3175	.3221	.3456	.3268	.3535	.6884	.4220	.5018	.4070	.6884
glass2	.2023	.2083	.2269	.2087	.2269	.4701	.1938	.2249	.2390	.4701
heart	.1556	.1593	.1371	.1741	.1667	.4635	.1550	.1847	.2180	.1484
hepatitis	.1125	.1250	.1750	.1500	.1250	.1877	.1294	.1302	.1750	.1877
iris	.0533	.0533	.0800	.0667	.0733	.0563	.0485	.0763	.0400	.0427
letter	.1433	.2076	.1511	.2520	.1712	.3530	.3068	.1752	.1220	.3092
lymphography	.2078	.2097	.3453	.1452	.2775	.2794	.1470	.1784	.2160	.3624
mofn-3-7-10	.0000	.0000	.0830	.1357	.0908	.1328	.1367	.0000	.1600	.1328
pima	.2427	.2394	.2916	.2737	.2606	.2569	.2505	.2384	.2590	.2569
satimage	.1543	.1712	.1374	.1920	.1795	.2220	.1730	.1420	.1770	.2710
segment	.0415	.0510	.1364	.0740	.0500	.1364	.0701	.0571	.0820	.1130
shuttle-small	.0113	.0052	.0108	.0142	.0102	.0186	.0083	.0052	.0060	.1349
soybean-large	.0758	.0704	.3451	.0885	.0746	.3373	.0920	.0663	.0890	.6466
vehicle	.3276	.3246	.3154	.4573	.3452	.4478	.3453	.2727	.3170	.5077
vote	.0552	.0552	.0851	.0966	.0621	.0420	.0370	.0487	.0530	.0463
waveform	.1630	.1785	.2566	.1795	.2516	.3281	.1772	.2534	.3490	.4345
Average	.1412	.1506	.1837	.1734	.1640	.2314	.1613	.1554	.1676	.2409

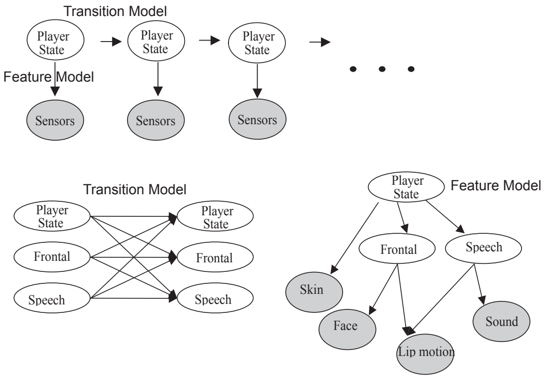


Fig. 8. DBN structure for Smart Kiosk dataset

We believe that the primary advantage of our approach is its simplicity and computational efficiency, coupled with its good performance in practice. Its use of weighted maximum likelihood parameter learning uniquely determines the parameters of the Bayesian network, providing an efficient mechanism for discriminative training.

D. Experiments on Boost-DBN

We used two real-world time-series data in our experiments. The first data set is taken from the Smart Kiosk project by Choudhury et al. [8] [40]. Smart Kiosk is a open-mike speech interface for a Black Jack game. The kiosk has a microphone and a camera input to retrieve visual and audio cues. Visual cues include the detection of skin, face and lip-motion to sense human presence. A simple audio cue is obtained by monitoring excursions in the audio signal above and below its moving average. In addition to the sensors, we use the state of the game as an auxiliary feature. In the training data, all state and features are binary and observed. The goal is to recover the state of the player (i.e. speaking or not) at each time slice given a sequence of observed sensory data.

Figure 8 illustrates the topology of the DBN for the Kiosk experiment. In addition to the sensory node and player state node, we added two hidden state variables named $Frontal$ and $Speech$, each is the parent node of related sensors. The intermediate state variables define meta-features which are formed by a combination of attributes.

The second data set is a collection of 37 sequences of multi-part FAQs, collected from various newsgroups. This dataset was previously used by McCallum et al. [35]. Each time-sequence contains 1000 to 2500 data samples. Each sample corresponds to one sentence in the article and contains 20 binary features and a state label for the topic of the sentence (introduction, question, answer or conclusion). For this experiment, the goal is to accurately recover the state label sequence given a instantiation of feature sequences. Since this is a 4 class problem, we used Adaboost.MH algorithm [17] on top of DBN.

The DBN topology for this problem is given in Figure 9. For tractability, we used naive Bayes as feature model. All experiments were done using N fold cross validation, where N is the total number of sequence available for particular dataset.

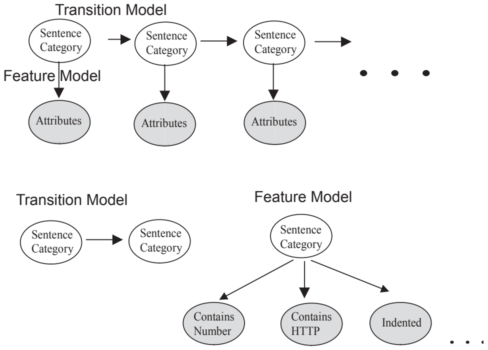


Fig. 9. DBN structure for FAQ dataset

Name	DBN	Boosted DBN	CRF
kiosk	0.7490	0.9120	**0.9641**
FAQ1(fetish)	0.8330	**0.9156**	0.8300
FAQ2(genetic)	0.9242	**0.9464**	0.8756
FAQ3(general)	0.8750	0.8960	**0.9015**
FAQ4(aix)	0.8767	**0.9047**	0.8663
FAQ5(bsd)	0.7981	**0.8174**	0.8033
FAQ6(neural)	0.9049	**0.9173**	**0.9129**
FAQ7(acorn)	0.8870	**0.8884**	**0.8882**

TABLE V

Testing error for DBN, Boosted DBN and CRF. For each dataset, the algorithm with the best classification accuracy is highlighted.

Table V lists the average label classification accuracy for Boosted-DBN together with DBN and Conditional Random Field (CRF) [31]. CRF fits an exponential model to the conditional distribution of the state labels given the attributes, and is generally considered as the state-of-the-art discriminative model for label sequence prediction task. We provide a more detailed description of CRF in the related work section.

As shown in Table V, Boosted-DBN significantly outperforms DBN on the Kiosk dataset and moderately outperforms DBN on the FAQ datasets. It is reasonable to conclude that Boosted-DBN is an effective method to improve the label sequence prediction accuracy.

Also, Boosted-DBN has comparable classification accuracy as CRF. Boosted-DBN slightly outperforms CRF in 4 datasets, while CRF outperforms Boosted-DBN in 2 and they tie in 2. However, in our experiments, Boosted-DBN has much faster convergence rate than CRF. While CRF takes more than 400 iteration before convergence, Boosted-DBN takes only 10-30 iterations to get good classification accuracy. Since each iteration of CRF training has roughly the same computational complexity as one boosting iteration for Boost-DBN, Boost-DBN has less training computational complexity.

IX. Related Work

This work is an extension and generalization of our previous works [8] [39] [27]. In [8] [39], we empirically showed that in the task of audio-visual sensor integration, AdaBoost improves the classification accuracy of Dynamic Bayesian Network. [27] introduced BAN algorithm as an efficient discriminative structure learning mechanism for the training of static Bayesian network. This paper unifies two algorithms by proposing a Boosted Bayesian Network Classifier framework and an interpret it as a graphical model. We also provide a more detailed theoretical analysis and a more complete evaluation, particularly on the dynamic Bayesian network.

Elkan [16] demonstrated the excellent classification performance of boosted Naive Bayes and pointed out its efficiency training mechanism. We build on this work by extending the use of boosting to structure learning. In contract to the experiments in [16], we include a more thorough comparison between BNB and a wide variety of competing methods on large set of standard datasets. Greiner and Zhou [23] proposed the ELR algorithm to directly maximize the CLL score of Bayesian network via gradient descent and line search. Therefore ELR-NB is essentially a logistic regression model. Their results and those of [24] all support our observation that discriminative training methods over Bayesian networks are preferred when the original model structure is incorrect. BNC algorithm constructs a Bayesian network by greedy search for structures that maximize a discriminative criteria. Schneiderman [48] used approach similar to BNC to discriminatively learn a restricted Bayesian network structure for object detection.

Maximum Entropy Markov Model (MEMM) [35] and Conditional Random Field (CRF) [31] are two alternative discriminative models for temporal sequence classification. During parameter learning, MEMM fits an exponential model to the state variable given previous state and the observation using Generalized Iterative Scaling [12].

However, as pointed out by Lafferty et al. [31], MEMM suffers from the label-bias problem. MEMM tends to ignore the attribute value in the presence of a sparse transition table. CRF avoids label bias problem by maximizing the CLL score for the entire state sequence given the attribute sequence via Improved Iterative Scaling [4]. Altun et al. [1] proposed to optimize an exponential loss function an alternative method to train CRF model. Boost-DBN is more simpler and more efficient to train, while have comparable performance as CRF. Furthermore, since Boost-DBN is not a finite state model, it does not suffer from label-bias problem as MEMM does.

X. Summary and Conclusion

This paper proposes an Boosted Bayesian Network Classifier framework to unify our previous work to improve the classification accuracy on both static and dynamic Bayesian networks. We proposed an interpretation of Boosted Bayesian network classifier as a graphical model consisting of a collection of Ensemble Bayesian network models.

We also proposed BAN algorithm, an efficient structure learning algorithm that generalizes Boosted Naive Bayes, and demonstrated that BAN can further improve the classification accuracy of BNB and significantly outperforms competing discriminative methods including TAN, BNC-2P and BNC-MDL. We also demonstrated that BNB and BAN are more efficient to train than ELR algorithm while having comparable accuracy.

Furthermore, we expanded the previous work [8] on Boost-DBN algorithm to include a more detailed theoretical analysis, and conducted a comprehensive empirical experiments on the Boosted-DBN model in both sensor fusion and part of speech tagging task. We demonstrated that Boost-DBN model improves the classification accuracy of HMM, and have comparable performance as Conditional Random Field, but with significant faster convergence rate.

We believe the primary advantage of Boosted Bayesian Network classifiers is their implementation simplicity, efficient training algorithm and fast convergence rates. Coupled with competitive classification accuracy against other more complex discriminative methods, we believe Boosted Bayesian Network classifiers are a collection of worthwhile tools for the machine learning community.

XI. Acknowledgements

The authors would like to thank Matt Mullin for several fruitful discussions and for suggesting the chain-structured model in Figure 5. We also would like to thank Minyoung Kim for providing an implementation of CRF and helping to conduct experiments on the FAQ datasets.

This material is based upon work supported in part by the National Science Foundation under NSF Grant IIS-0205507.

References

[1] Y. Altun, T. Hofmann, and M. Johnson. Discriminative learning for label sequences via boosting. In *Proc. 15th Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 977–984, 2003.

[2] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden Markov support vector machines. In *Proc. 20th International Conference on Machine Learning (ICML)*, Washington DC, 2003.

[3] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139, 1999.

[4] A. Berger. The improved iterative scaling algorithm: A gentle introduction.

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

[6] C. Chelba and A. Acero. Conditional maximum likelihood estimation of naive Bayes probability models using rational function growth transform. Technical Report MSR-TR-2004-33, Microsoft, 2004.

[7] D. M. Chickering and D. Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29(2-3):181–212, 1997.

[8] T. Choudhury, J. M. Rehg, V. Pavlović, and A. Pentland. Boosting and structure learning in dynamic Bayesian networks for audio-visual speaker detection. In *Proc. 16th International Conference on Pattern Recognition*, 2002.

[9] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

[10] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[11] D. Cutting, J. Kupiec, J. Pedersen, and P. Sibun. A practical part-of-speech tagger. In *Proc. 3rd conference on Applied natural language processing*, pages 133–140, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[12] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480, 1972.

[13] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.

[14] H. Drucker and C. Cortes. Boosting Decision Trees. In *Proc. 8th Advances in Neural Information Processing Systems (NIPS)*, pages 470–485. 1996.

[15] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience Publication, New York, 1973.

[16] C. Elkan. Boosting and naive Bayesian learning. Technical report, Department of Computer Science and Engineering, University of California, San Diego., 1997.

[17] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 38(2):337–374, 2000.

[18] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[19] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proc. 16th International Joint Conf. on Artificial Intelligence (IJCAI)*, pages 1300–1309, 1999.

[20] N. Friedman and D. Koller. Being Bayesian about network structure. In *Proc. 16th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 201–210, June 2000.

[21] N. Friedman, K. Murphy, and S. Russell. Learning the structure of dynamic probabilistic networks. In *Proc. 14th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 139–147, San Francisco, 1998.

[22] Z. Ghahramani. Learning dynamic Bayesian networks. *Adaptive Processing of Sequences and Data Structures . Lecture Notes in Artificial Intelligence*, 1387:168–187, 1998.

[23] R. Greiner and W. Zhou. Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers. In *Proceedings of annual meeting of the American Association for Artificial Intelligence*, pages 167–173, 2002.

[24] D. Grossman and P. Domingos. Learning Bayesian network classifiers by maximizing conditional likelihood. In *Proc. 21st International Conference on Machine Learning (ICML)*, pages 361–368, Banff, Canada, 2004. ACM Press.

[25] D. Heckerman. A tutorial on learning with bayesian networks. Technical Report MSR-TR-95-06, Microsoft Reserach, 1995.

[26] D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[27] Y. Jing, V. Pavlović, and J. M. Rehg. Efficient discriminative learning of Bayesian network classifiers via boosted augmented naive Bayes. In *[Accepted for publication]Proc. 22nd International Conf. on Machine Learning (ICML)*, 2005.

[28] V. Jojic, N. Jojic, C. Meek, D. Geiger, A. Siepel, D. Haussler, and D. Heckerman. Efficient approximations for learning phylogenetic HMM models from data. *Bioinformatics*, 20(1):161–168, 2004.

[29] E. Keogh and M. Pazzani. Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In *Proc. 7th International Workshop on Artificial Intelligence and Statistics*, pages 225–230, 1999.

[30] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[31] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning (ICML)*, pages 282–289, 2001.

[32] W. Lam and F. Bacchus. Learning Bayesian belief networks. an approach based on the mdl principle. *Computational Intelligence*, 10:269–293.

[33] P. Langley, W. Iba, and K. Thompson. An analysis of Bayesian classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 223–228, San Jose, CA, 1992. AAAI Press.

[34] U. Lerner, B. Moses, S. Maricia, S. McIlraith, and D. Koller. Monitoring a complez physical system using a hybrid dynamic Bayes net. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence (UAI-02)*, pages 301–310, San Francisco, CA, 2002. Morgan Kaufmann Publishers.

[35] A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proc. 17th International Conference on Machine Learning (ICML)*, pages 591–598, San Francisco, CA, 2000.

[36] K. Murphy. The Bayes net toolbox for matlab. *Computing Science and Statistics*, 33, 2001.

[37] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In *Proc. 14th Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 841–848, 2002.

[38] V. Pavlović, A. Garg, and S. Kasif. A Bayesian framework for combining gene predictions. *Bioinformatics*, 18(1):19–27, 2002.

[39] V. Pavlović, A. Garg, and J. M. Rehg. Boosted learning in dynamic Bayesian networks for multimodal speaker detection. *Proceedings of the IEEE*, 91(9):1355–1369, 2003.

[40] V. Pavlović, A. Garg, J. M. Rehg, and T. Huang. Multimodal speaker detection using error feedback dynamic Bayesian networks. In *2000 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 34–41, 2000.

[41] V. Pavlović, J. M. Rehg, T. Cham, and K. P. Mu rphy. A dynamic bayesian network approach to figure tracking using learned dy namic models. In *Intl. Conf. on Computer Vision*, 1999.

[42] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[43] L. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[44] J. M. Rehg, V. Pavlović, T. S. Huang, and W. T. Freeman. *Special Section on Graphical models in Computer Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7), 2003.

[45] G. Ridgeway, D. Madigan, T. Richardson, and J. O'Kane. Interpretable boosted naive Bayes classification. In *Proc. 4th International Conference on Knowledge Discovery and Data Mining*, 1998.

[46] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Proc. 11th Annual Conference on Computational Learning Theory*, 1998.

[47] R. E. Schapire and Y. Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[48] H. Schneiderman. Learning a restricted Bayesian network for object detection. In *Proc. Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 639–646, June 2004.

[49] E. Segal, R. Yelensky, and D. Koller. Genome-wide discovery of transcriptional modules from dna sequence and gene expression. *Bioinformatics*, 19(1):273–82, 2003.

[50] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Proc. 16th Conference on Advances in Neural Information Processing Systems (NIPS)*, Cambridge, MA, 2004. MIT Press.

[51] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 762–769, Washington, DC, June 2004.

[52] G. Webb and Z. Zheng. Multistrategy ensemble learning: Reducing error by combining ensemble learning techniques. *IEEE Transactions on Knowledge and Data Engineering*, 16(8):980–991, August 2004.

[53] M. Zou and S. D. Conzen. A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. *Bioinformatics*, 21(1):71–79, 2005.