Discriminant Parameter Learning of Belief Net Classifiers

Russell Greiner and **Wei Zhou***
Department of Computing Science
University of Alberta
Edmonton, AB T6G 2E8 Canada
{ wei, greiner }@cs.ualberta.ca

June 25, 2001

Abstract

Recent results have shown that "Bayesian classifiers", including NaïveBayes, perform extremely well as classifiers. Essentially all of the associated learners seek the parameters that maximize the *likelihood* of the sample. As these parameters may not maximize the classification accuracy, we propose instead finding the parameters that maximize *conditional likelihood*, of the class label given the instance description, over the training sample. This paper first formally describes this "discriminant learning" task and analyzes its inherent complexity. Based on this analysis, we present an effective gradient-descent algorithm DEP for learning the best (read "most accurate") parameters for any given belief net structure, from either complete or incomplete training data. We then provide empirical evidence that DEP effectively produces accurate classifiers — often competitive with the best state-of-the-art classifiers, and in certain situations, more accurate than the standard learning algorithms. We show that this is especially true in the common situation where the BN-structure is incorrect; *e.g.*, when learning NaïveBayes classifiers.

Keywords: Naive Bayes, Classification Performance, Discriminant Learning, (Bayesian) Belief Nets

1 Introduction

Many tasks — including fault diagnosis, pattern recognition and forecasting — can be viewed as *classification*, as each requires assigning the class ("label") to a given instance, which is typically specified by a set of attributes. There are a number of learners that try to learn accurate classifiers from labeled instances, using a variety of representations for these classification functions, including decision trees, rules sets, and neural nets [Mit97]. While (Bayesian) belief nets (BNs) [Pea88] are known to be useful tools for representing and reasoning about uncertain information, they were not considered as classifiers until the observation [LIT92] that it was easy to learn very accurate "NaïveBayes classifiers" [DH73], which are trivial BNs whose attributes are conditionally independent given the classification.

This success has led to a number of effective "Bayes classifiers", each corresponding to some special type of belief net (BN) structure, including Tree-Augmented Networks, BN-augmented networks, feature-selection NaïveBayes, etc. [FGG97, KS96, CG99]. Each of the associated learning algorithms first learns a good network structure from this class (if necessary), then seeks the

*Authors listed alphabetically.

parameters appropriate for this structure. Essentially all of these learners use the parameters that maximize the *likelihood* of the training sample [Hec98, Bun96]. This approach does produce BNs that are good models of the distribution; moreover, the associated parameter-learning subroutine is incredibly efficient [CH92, Hec98]. However, as explained below, these parameters need not maximize the *classification accuracy*, which is the relevant objective function.

This paper pursues an obvious alternative: explicitly seek the most accurate *classifier*, rather than the closest *distribution* — *i.e.*, discriminative rather than generative learning [Bis98, Vap98]. We implement this by seeking the parameters (for the fixed structure) that maximize the *conditional* likelihood, of the class label c_i given the instance e_i; this differs from the standard (non-discrimant) approach of finding the parameters that maximize the likelihood of the labeled instances $\langle e_i, c_i\rangle$. We demonstate below that this approach works effectively over a wide variety of situations.

Section 2 provides the foundations — overviewing belief nets, then motivating our specific approach: learning the parameters (for a fixed belief net structure) that maximize *conditional* likelihood. After Section 3 shows this task to be NP-hard, it then provides a gradient-descent learning algorithm, DEP.

Section 4 reports empirical results which demonstrate that DEP performs well over a variety of situations; and in certain situations, better than the standard learning algorithms that maximize likelihood. (1) We first consider the simple case of learning accurate NaïveBayes classifiers from complete data. Our study, over a number of datasets, shows that the classifiers obtained using DEP are typically more accurate than the ones obtained using the standard "observed frequency estimate" (OFE) approach [CH92]. We show, moreover, that our NaïveBayes+DEP classifiers are competitive with the best learners, BN-based or not. (2) We next consider learning from *partially specified* instances, and show that DEP is comparable to (and occasionally superior to) the standard algorithms for this situation: expectation-maximization "EM" [Hec98] and adaptive probabilistic networks "APN" [BKRK97]. (3) Finally, we include a simple example to reinforce the common wisdom that the advantage of our discriminant learner DEP over generative learners (OFE for complete data and APN/EM for incomplete data) comes from DEP's ability to cope with incorrect BN-structures. Section 5 discusses these results, and related issues. Section 6 surveys the relevant literature, indicating how this works differs from other work on learning belief nets, as well as the body of work on discriminant learning in general. The appendix presents proofs of the theoretical claims made in the paper.

2 Framework

A (Bayesian) belief net (BN), $B = \langle \mathcal{V}, \mathcal{A}, \Theta\rangle$ is a directed acyclic graph whose nodes each represent a random variable, and whose arcs $a = \langle R, F\rangle \in \mathcal{A}$ each represent a probabilistic dependency between R and F. There is also a conditional probability distribution (CPtable) $\theta_R \in \Theta$ associated with each node $R \in \mathcal{V}$ that describes R's posterior distribution, as a function of the values of its (immediate) parents $\mathbf{F} \subset \mathcal{V}$; in particular, the parameter $\theta_{r|\mathbf{f}}$ represents the network's term for $P(\, R = r \mid \mathbf{F} = \mathbf{f}\,)$ [Pea88].

As a belief net represents a joint distribution over the variables $\mathcal{V}$, it can be used to determine the conditional probability $P(\, C \mid \mathbf{E} = \mathbf{e}\,)$ of any variable C, conditioned on any assignment to some set of other variables $\mathbf{E} = \mathbf{e}$. As we are dealing with the classification task, we will consider C to be the class variable, and (if the data is complete) $\mathbf{E} = \mathcal{V} - \{C\}$. The classifier $class_B(\cdot)$ for the BN B will "label" an instance $\mathbf{E} = \mathbf{e}$ with the most likely class $class_B(\mathbf{E} = \mathbf{e}) = \text{argmax}_c\{P(\, C = c \mid \mathbf{E} = \mathbf{e}\,)\}$.

A good classifier is one that produces the appropriate answers to these unlabeled instances as often as possible. We use the standard "classification error" (aka "0/1" loss) to evaluate the resulting B-based classifier $class_B$

$$\mathrm{err}(\,B\,) \quad = \quad P_{\langle \mathbf{e},c\rangle}(\,class_B(\mathbf{e}) \neq c\,) \tag{1}$$

over the true distribution of labeled instances $\langle \mathbf{e}, c\rangle$,[1] which we will approximate with the empirical score over a given labeled sample S:

$$\widehat{\mathrm{err}}^{(S)}(\,B\,) \quad = \quad \frac{1}{|S|} \sum_{\langle c,\mathbf{e}\rangle \in S} [\![\, class_B(\mathbf{e}) \neq c \,]\!] \tag{2}$$

where $[\![\, A \neq B \,]\!]$ is 1 when $A \neq B$ and 0 otherwise.

Our goal is a belief net $B^* = \langle \mathcal{V}, \mathcal{A}, \Theta^* \rangle$ with the given structure $\langle \mathcal{V}, \mathcal{A} \rangle$, that minimizes the error with respect to the true distribution. That is, we want an learner that takes as input a belief net structure $G = \langle \mathcal{V}, \mathcal{A} \rangle$, and a sample of labeled instances $S = \{\langle e_i, c_i \rangle\}_i$, and returns the CPtable entries for G that produce the smallest empirical score $\widehat{\mathrm{err}}^{(S)}(\,\cdot\,)$ score (Equation 2). Given a sufficiently large large sample, this will correspond to the BN with the best $\mathrm{err}(\,\cdot\,)$ score.

Conditional Likelihood: The expected error of a classifier corresponds to the conditional probability — *e.g.*, if $P(\,\texttt{Cancer} = \texttt{true} \,|\, \texttt{Smoker} = \texttt{true}\,) = 0.8$, then we expect the associated classifier, which labels the instance "$\langle \texttt{Smoker} = \texttt{true};\ \texttt{Cancer}\rangle$ as "True", will be wrong 20% of the time — *i.e.*, have an error of 20%. In general, the *expected* error of a classifier, for each instance, is $\widehat{\mathrm{err}}^{(E=e)}(\,B\,) = 1 - P_B(\,H = h \,|\, E = e\,)$. This suggests that minimizing error corresponds to maximizing the conditional likelihood of that instance $P(\,H = h \,|\, E = e\,)$. As log() is a monotonic function, this is equivalent to maximizing the *log* of the conditional likelihood.

We therefore define the "(negative) log conditional likelihood" of a belief net B, over the distribution of labeled instances, as

$$\mathrm{LCL}_P(\,B\,) \;=\; -\sum_{\langle \mathbf{e},c\rangle} P(\,\mathbf{e}\,)\ \log P_B(\,c \,|\, \mathbf{e}\,) \tag{3}$$

where $P_B(\,\chi\,)$ corresponds to the value the "distribution" B assigns to the event χ. Given a sample S, we can approximate this as

$$\widehat{\mathrm{LCL}}^{(S)}(\,B\,) \quad = \quad -\frac{1}{|S|} \sum_{\langle c,\mathbf{e}\rangle \in S} \log(\,P_B(\,c \,|\, \mathbf{e}\,)\,) \tag{4}$$

([FGG97] similarly note that minimizing Equation 3 is equivalent to minimizing the *conditional cross-entropy*, which means the model that minimizes this LCL-scoring function should be a good classifier.) Note also that many research projects, including [BKRK97], use this measure when evaluating their BN classifiers.

While this $\widehat{\mathrm{LCL}}^{(S)}(\,B\,)$ formula closely resembles the "log likelihood" function

$$\widehat{\mathrm{LL}}^{(S)}(\,B\,) \quad = \quad -\frac{1}{|S|} \sum_{\langle \mathbf{e},c\rangle \in S} \log(\,P_B(\,\mathbf{e}, c\,)\,) \tag{5}$$

used by many BN-learning algorithms, there are some critical differences: [FGG97] note

$$\widehat{\mathrm{LL}}^{(S)}(\,B\,) \quad = \quad -\frac{1}{|S|} \left[\sum_{\langle c,\mathbf{e}\rangle \in S} \log(\,P_B(\,c \,|\, \mathbf{e}\,)\,) \;+\; \sum_{\langle \mathbf{e},c\rangle \in S} \log(\,P_B(\,\mathbf{e}\,)\,) \right]$$

[1] Following standard convention, we will assume that the "distribution over instances" corresponds to the underlying tuple distribution, from which we derive $P(\,C \,|\, \mathbf{E}\,)$; and so we will use $P(\,\cdot\,)$ to refer to both; see [GGS97].

where the first summation resembles our $\widehat{\mathrm{LCL}}^{(\cdot)}(\ )$ score, which measures how well our network will answer the relevant queries, while the second summation is irrelevant to our task. In particular, it is easily possible that $\widehat{\mathrm{LL}}^{(S)}(B_\alpha) < \widehat{\mathrm{LL}}^{(S)}(B_\beta)$ but $\widehat{\mathrm{LCL}}^{(S)}(B_\alpha) > \widehat{\mathrm{LCL}}^{(S)}(B_\beta)$. Here, a learner that uses the $\widehat{\mathrm{LL}}^{(S)}(\cdot)$ criterion will prefer B_α over B_β, even though B_β is a better classifier. (Section 5 provides other arguments explaining why our approach may work than the $\widehat{\mathrm{LL}}^{(S)}(\cdot)$-based approaches; and Section 6 surveys other relevant literature.)

3 Learning Algorithm

As explained above, we are seeking the CPtable entries that have the optimal conditonal likelihood, wrt the data. Unfortunately, this task is intractable:

Theorem 1 *It is NP-hard to find the values for the CPtables of a fixed BN-structure that produce the smallest (empirical)* $\widehat{LCL}^{(S)}(\cdot)$*-score for a given sample* S. ∎

To sidestep this problem, we defined a simple gradient-descent algorithm, DEP (for "Discriminant Estimation of Parameters), that attempts to improve the empirical score $\widehat{\mathrm{LCL}}^{(S)}(B)$ by changing the values of each CPtable entry $\theta_{r|\mathbf{f}}$. Following [BKRK97], we first built the DEP$_\theta$, system that considered the Θ space: On each iteration, it first climbed down the gradient *with respect to each* θ_i, then corrects the resulting $\theta_{r|\mathbf{f}} \in \Theta$ values to insure that $\theta_{r|\mathbf{f}} \geq 0$ and $\sum_r \theta_{r|\mathbf{f}} = 1$.

We next built a system DEP = DEP$_\beta$ that avoids this complication by using a different set of parameters — "$\beta_{r|\mathbf{f}}$" — where each

$$P_B(R{=}r \mid \mathbf{F}{=}\mathbf{f}) \;=\; \theta_{r|\mathbf{f}} \;=\; e^{\beta_{r|\mathbf{f}}} / \sum_{r'} e^{\beta_{r'|\mathbf{f}}}$$

As the $\beta_{r|\mathbf{f}}$'s sweep over the reals, the corresponding $\theta_{r_i|\mathbf{f}}$'s will satisfy the appropriate constraints. (These are also called the "softmax" parameters [Bis98].[2])

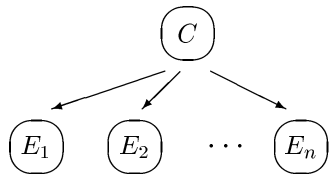
Given a set of labeled queries, DEP descends in the direction of the total derivative wrt these queries, which of course is the sum of the individual derivatives:

Proposition 2 *For a single labeled instance* $[\mathbf{e}; c]$*:*

$$\frac{\partial\, \widehat{LCL}^{([\mathbf{e};c])}(B)}{\partial\, \beta_{r|\mathbf{f}}} \;=\; [P_B(r,\mathbf{f} \mid \mathbf{e}, c) - P_B(r,\mathbf{f} \mid \mathbf{e})] \;-\; \theta_{r|\mathbf{f}}\,[P_B(\mathbf{f} \mid \mathbf{e}, c) - P_B(\mathbf{f} \mid \mathbf{e})]\ .$$

We then incorporated several enhancement to speed-up this computation, including standard ideas of line-search and conjugate gradient [Bis98]. Another important improvement stems from the observation that this derivative is 0 whenever R is independent of C given $\mathbf{E}$ — which makes sense, as this condition means that the $\theta_{r|\mathbf{f}}$ term plays no role in computing $P_B(c \mid \mathbf{e})$; see [BKRK97]. This led to very significant savings in some situations (but, of course, not in the complete-data case; see below.)

[2]Note that these parameters can be viewed as the natural parameterization in an exponential family representation for conditional distributions [Thi95].



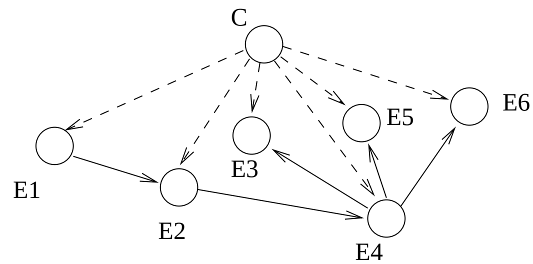


Figure 1: (a) NaïveBayes Structure; (b) TAN structure (see [FGG97, Fig 3])

4 Empirical Exploration

The DEP algorithm takes, as arguments, a BN-structure G and a dataset $S = \{\langle \mathbf{e}_i, c_i \rangle\}_i$, and returns a value for each parameter $\theta_{r|\mathbf{f}} \in \Theta$. To explore its effectiveness, we compared the $\mathrm{err}(\cdot)$ performance of the resulting Θ_{DEP} with the results of other algorithms that similarly learn CPtable values for a given structure.

We say the data is "complete" if every instance specifies a value for every attribute; hence "$E_1 = e_1, \ldots, E_n = e_n$" is complete (where $\{C, E_1, \ldots, E_n\}$ is the full set of variables) but "$E_2 = e_2, E_7 = e_7$" is not. When the data is complete, we compare DEP to the standard "observed frequency estimate" (OFE) approach, which is known to produce the parameters that maximize likelihood for a given structure [CH92]. That is, assume C is a parent of E_1, as in Figure 1(a), and let $\#[\,C = 1\,]$ (resp., $\#[\,E_1 = 0, C = 1\,]$) be the number of instances in the dataset where $C = 1$ (resp., $E_1 = 0$ and $C = 1$). Then OFE sets the value of the $\theta_{E_1=0|C=1}$ parameter as:

$$\theta_{E_1=0|C=1} \;:=\; \frac{\#[\,E_1 = 0, C = 1\,]}{\#[\,C = 1\,]} \tag{6}$$

(Some learners instead use the Laplacian corrected version, $\theta_{E_1=0|C=1} \;:=\; \frac{\#[\,E_1=0,C=1\,]+1}{\#[\,C=1\,]+2}$ assuming E_1 is binary; see [Hec98].)

When the data is incomplete, we compare DEP to the standard expectation-maximization algorithm EM [Hec98] and to the gradient-ascent algorithm APN [BKRK97].[3]

We present only the results of the DEP (that is, DEP_β) algorithm, as we found its performance strictly dominated DEP_θ's. Also, as we are concerned with classification performance, this paper focuses on classification error; where appropriate we will also present the results wrt mean-squared-error:

$$\widehat{mse}^S(\,B\,) \quad = \quad \sum_{\langle c,\mathbf{e}\rangle \in S} P(\,\mathbf{e}\,) \;\times\; [P_B(\,c\,|\,\mathbf{e}\,) - P(\,c\,|\,\mathbf{e}\,)]^2 \tag{7}$$

In either case, we use hold-out or cross-validation datasets.

4.1 NaïveBayes — Complete, RealWorld Data

Our first experiments dealt with the simplest situation: learning the NaïveBayes parameters from complete data. Recall that the NaïveBayes structure requires that the attributes are independent given the class label — see Figure 1(a).

Here, we compared the relative effectiveness of DEP with various other classifiers, over the same 25 datasets that [FGG97] used for their comparisons: 23 from UCIrvine repository [BM00],

[3] While the original APN_θ [BKRK97] climbed in the space of parameters θ_i, we instead used a modified APN_β system that uses the β_i values, as discussed in Section 3, as we found it worked better.

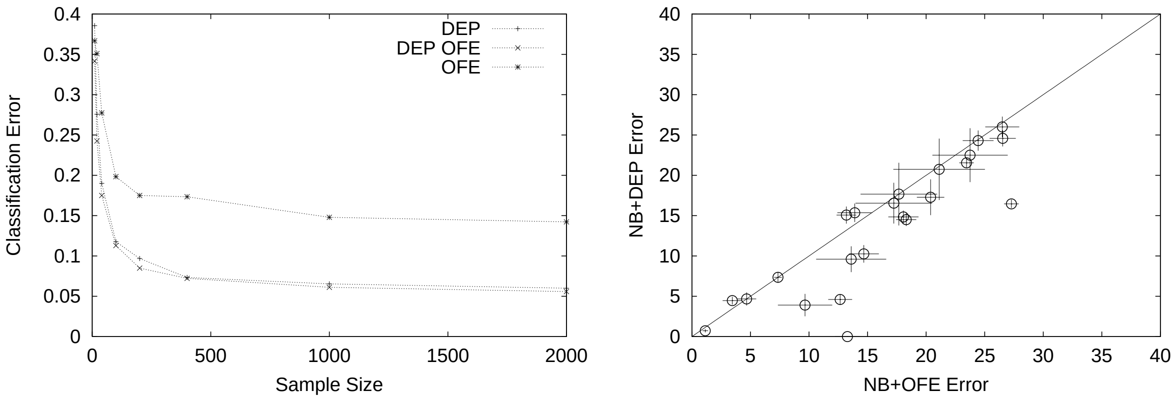


Figure 2: Comparing NB+DEP with NB+OFE: (a) learning curves for CHESS dataset; (b) over 25 datasets

plus "MOFN-3-7-10" and "CORRAL", which were developed by [KJ97] to study feature selection. To deal with continuous variables, we implemented supervised entropy discretization [FI93]. Our accuracy values were based on 5-fold cross validation for small data, and the holdout method for large data [Koh95]. As our data and testing procedure followed [FGG97], we can compare DEP system against the other classifiers they considered; see below. Table 1 summarizes the datasets, and our testing splits.

We use the CHESS dataset (36 binary or ternary attributes) to illustrate the basic behaviour of the algorithms. Figure 2(a) shows the performance, on this dataset, of our NB+DEP ("NaïveBayes structure + DEP parameter instantiation") system, versus the "standard" NB+OFE, which uses OFE to instantiate the parameters. We see that DEP is consistently more accurate than OFE, for any size training sample. We also see how quickly DEP converges to the best performance. The *ILQ-OFE* line corresponds to using OFE to initialize the parameters, then using the DEP-gradient-descent. (This is as opposed to beginning with some randomly initialized parameter values.) We see this has some benefit, especially for small sample sizes. All of our results, therefore, actually use this DEP-OFE.

Figure 2(b) provides a more comprehensive comparison, across all 25 datasets. (In the first set of scatter-plot figures, each point below the $x = y$ line is a dataset where NB+DEP was better than other approach — here NB+OFE. The horizontal and vertical lines around each point express the 1 standard-deviation error bars in each dimension.) As suggested by this plot, NB+DEP is significantly better than NB+OFE at the $p < 0.005$ level.[4] (We suggest why this may be true in Section 4.4.)

We also compared NB+DEP with other classifiers. A SNB+OFE classifier is produced by using OFE to instantiate a NaïveBayes structure that contains only a selected subset of the attributes [KJ97]; and C4.5 is a well-known decision-tree learner [Qui92]. As suggested by Figures 3(a) and 3(b), NB+DEP is significantly better than SNB+OFE (resp., slightly better than SNB+OFE) at the $p < 0.025$ level (resp., $p < 0.2$). (These SNB+OFE and C4.5 values are taken from [FGG97, Table 3]. Here we omitted three of the datasets where we obtained significantly different accuracy values on our common classifiers (NB+OFE, TAN+OFE), probably due to the different cross-validation partitioning of the small datasets, and/or different discretization implementation.)

[4]Each significance result reported is based on a 1-sided paired-t test [Mit97]. We report values up to $p < 0.2$; of course, values larger than $p < 0.05$ are probably not significant.

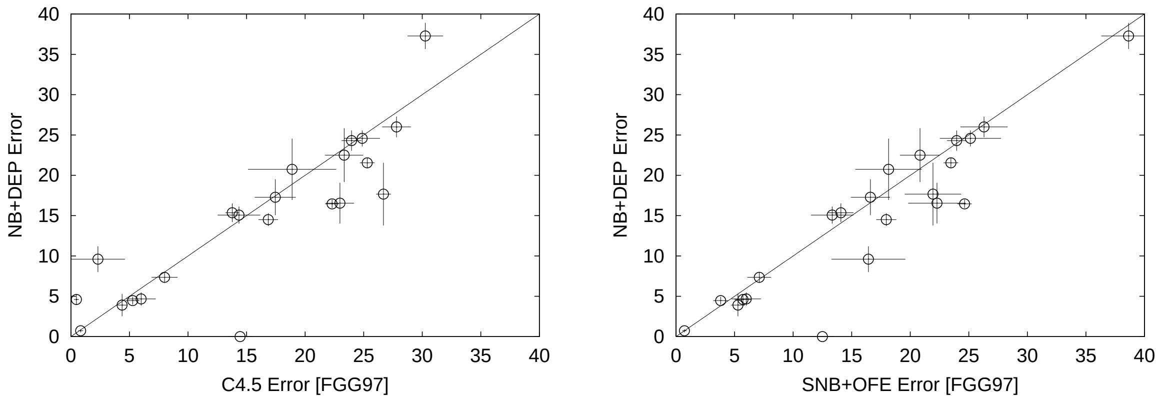


Figure 3: Comparing NB+DEP vs (a) SNB+OFE (b) C4.5

4.2 TAN — Complete, RealWorld Data

We next considered TAN ("tree augmented network") structures [FGG97]. In a TAN-structure, there is a link from the classification node down to each attribute. If we ignore those links, the remaining links, connecting attributes to each other, form a tree; see Figure 1(b). (Hence this representation allows each attribute to have at most one "attribute parent", and so this class of structures strictly generalize NaïveBayes.) [FGG97] provide an efficient algorithm for learning such TAN structures, given complete data.

Here, we first compared NB+DEP to TAN+OFE; the results appear in Figure 4. We see that DEP, even when handicapped with the simple NaïveBayes structure, performs about as well as OFE on TAN structures. (No significance, either way, using a paired-t test.) Of course, the limitations of the NaïveBayes structure may explain the poor performance of NB+DEP on some data. For example, in the artificial dataset CORRAL, as the class is a non-trivial function of the four relevant attributes, one must connect the four relevant attribute to predict the class. As NaïveBayes permits no such connection, all three NaïveBayes classifiers (NB+DEP, NB+OFE, SNB+OFE) perform poorly on this data. Of course, as TAN allows more expressive structures, it has a significant advantage here. It is interesting to note that our NB+DEP is still comparable to TAN+OFE, in general.

Would we do yet better by using DEP to instantiate TAN structures? Figure 5(a) shows that TAN+DEP does in fact work slightly better than NB+DEP, but only at the $p < 0.2$ level. Moreover, Figure 5(b) shows that TAN+DEP does consistently better than TAN+OFE — at a $p < 0.025$ level.

Notice that TAN+DEP actually did perfectly on the the CORRAL dataset, which NB+DEP found problematic. (The [FGG97, Table 3] paper compared TAN+OFE to NB+OFE, and to SNB+OFE. As we obtained similar results, we will not duplicate those figures; see Table 2.)

4.3 NaïveBayes — InComplete, RealWorld Data

All of the above studies used *complete* data. We next explored how well DEP could instantiate the NaïveBayes structure, using *incomplete* data.

Here, we used the datasets investigated above, but modified by randomly removing the value of each attribute, within each instance, with probability 0.25. (That is, this data is missing completely at random, MCAR [LR87].) We then compared DEP to the standard "missing-data" learning algorithms, APN and EM. In each case — for DEP, APN and EM — we initialize the parameters using the obvious variant of OFE that considers only the records that include values for the relevant

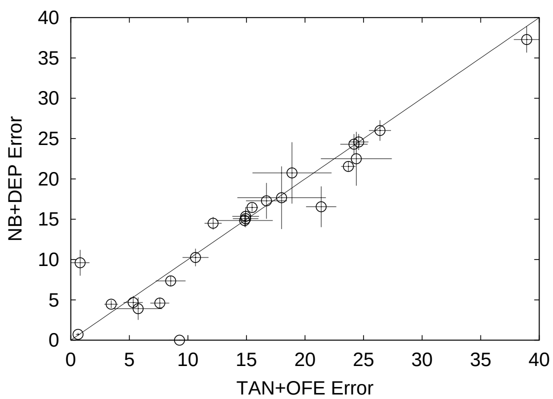


Figure 4: Comparing NB+DEP vs TAN+OFE

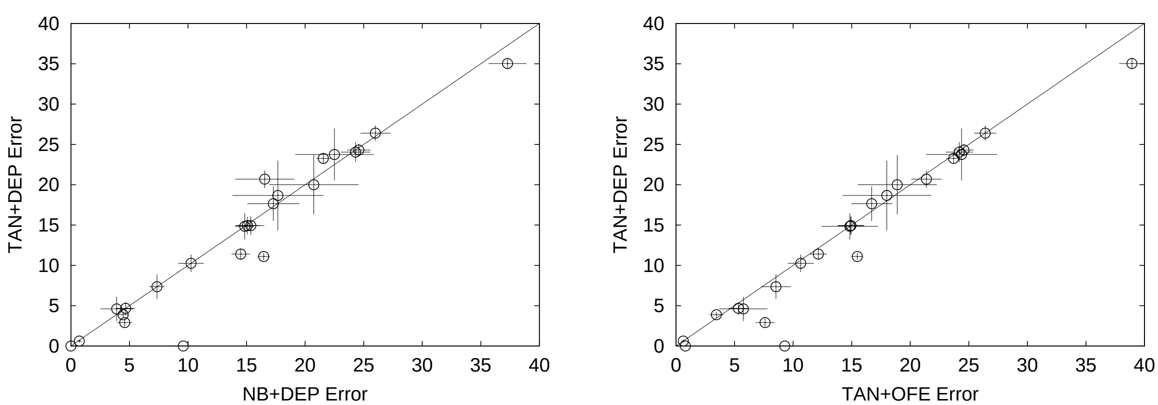


Figure 5: Comparing TAN+DEP with (a) NB+DEP; (b) TAN+OFE

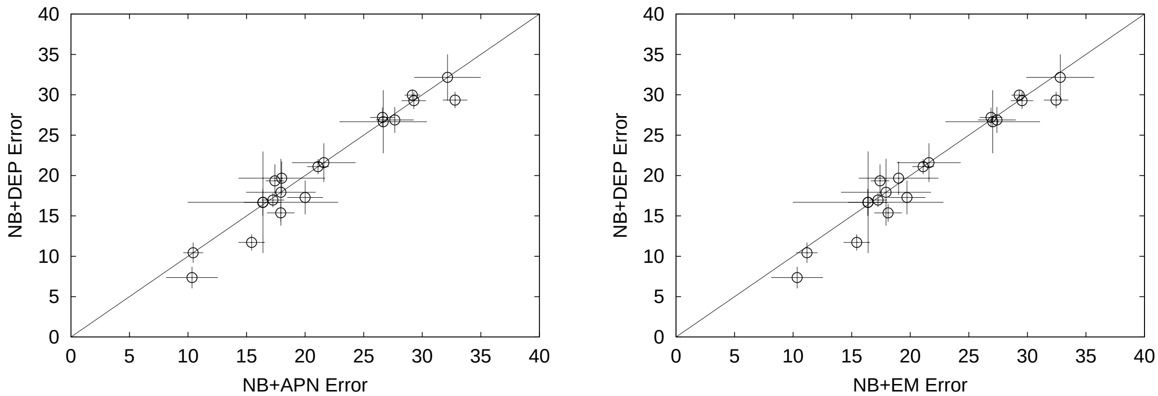


Figure 6: Comparing NB+DEP with (a) NB+APN; (b) NB+EM on Incomplete Data

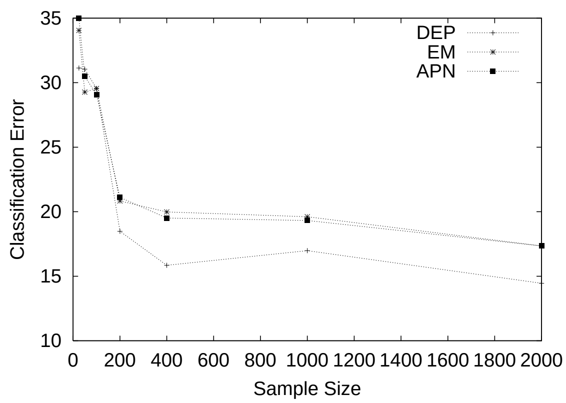


Figure 7: Comparing Parameter Learners, DEP vs EM,APN, for CHESS; incomplete data

node and all of its parents.

Here, we first learned the parameters for the NaïveBayes structure; Figure 7 shows the learning curve for the CHESS domain, comparing DEP to APN and EM. We see that DEP does better for essentially any sample size.

We also compared these algorithms over the rest of the 25 datasets; see Figures 6(a) and 6(b) for DEP vs APN and DEP vs EM, respectively. As shown, DEP does consistently better — in each case, at the $p < 0.025$ level.

Note that EM's first step corresponds to the obvious varient of OFE mentioned above: when computing the frequencies of an attribute A given the class C (to estimate the value of $\theta_{A|C}$), it will simply ignore a record if A was not specified. In this specific situation, EM will terminate after this single iteration, as these values actually maximize the likelihood of the data.[5]

Proposition 3 *When instantiating a NaïveBayes-structure from a data sample where the class variable is always specified but some attribute values are missing completely at random,* EM *will always*

- *converge after a single iteration,*
- *produce the maximum likelihood parameters.* ∎

[5]Of course, this claim is not true in general!

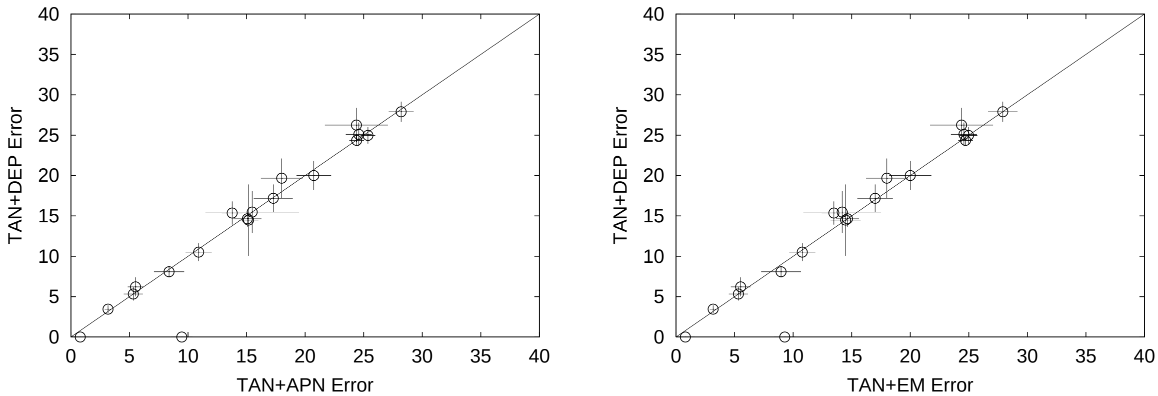


Figure 8: Comparing TAN+DEP with (a) TAN+APN; (b) TAN+EM on Incomplete Data

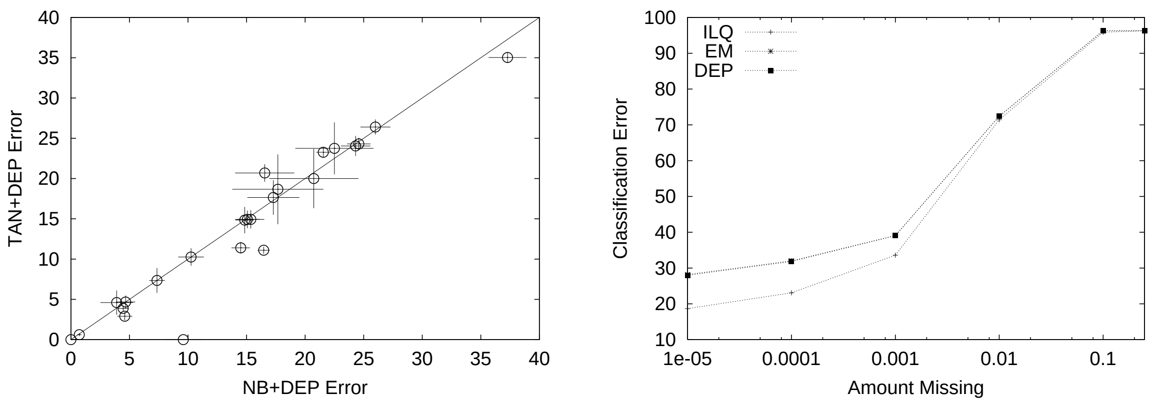


Figure 9: (a) Comparing TAN+DEP with NB+DEP on Incomplete Data (b) Comparing DEP to APN and EM on LETTER dataset, varying the quantity of missing data

We next tried to learn the parameters for a TAN structure. The standard TAN-learning algorithm computes the mutual information between each pair of attributes, conditioned on the class variable. This is straightforward when given complete information. Here, given *incomplete* data, we approximate mutual information between attributes A_i and A_j by simply ignoring the records that do not have values for both of these attributes. Our results, comparing TAN+DEP to TAN+APN and to TAN+EM, are shown in Figures 8(a) and 8(b) respectively. Here, we see that these systems are roughly equivalent: TAN+DEP is better than TAN+EM at $p < 0.1$, but it is not significantly better than TAN+APN.

Finally, we compared NB+DEP to TAN+DEP (Figure 9(a)), but found no significant difference.

Table 3 presents all of our empirical results related to missing data. (As we were concerned with the poor, but uniform, performance of all classifiers on the LETTER dataset, we further investigated their performances on this dataset, using various different missing-value rates. The results, shown in Figure 9(b), show that the error rate were not always so high (for smaller rates of missing data), and that various classifiers did have different performances when the missing data rate was smaller.)

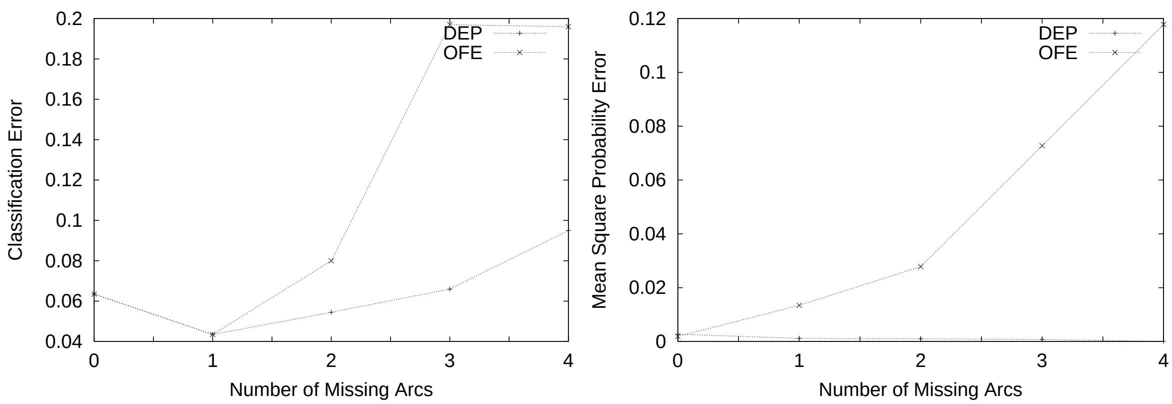


Figure 10: Comparing DEP to OFE, on Increasingly Incorrect Structures (NaïveBayes) using Complete Data, based on (a) Classification Error; (b) Mean Square Error

4.4 NaïveBayes — "Correctness of Structure" Study

It is well known that, in general, discriminant learning tends to be more robust to incorrect assumptions than generative learning, wrt the classification goal [CS89, Jor95, Rip96]. Here, this suggests our discriminant DEP should be more robust to incorrect structures than the generative learners — OFE, APN, EM.

Our earlier studies support this claim: The NaïveBayes structure, which makes the typically-incorrect assumption that the attributes are independent given the classification variable, is known to handicap generative learners, especially OFE [DP96]. The earlier studies demonstrated that DEP was not as handicapped here. We designed the following simple experiment to empirically investigate this claim, in general.

We used synthesized data, to allow us to vary the "incorrectness" of the structure. Here, we consider an underlying distribution, P_0 over the $k+1$ binary variables $\{C, E_1, E_2, \ldots, E_k\}$ where (initially)

$$P(\,C\,) = 0.9 \quad P(\,E_i \,|\, C\,) = 0.2 \quad P(\,E_i \,|\, \bar{C}\,) = 0.8 \tag{8}$$

and our queries were all complete; *i.e.*, each instance of the form $\vec{\mathbf{E}} = \langle \pm E_1, \pm E_2, \ldots, \pm E_k \rangle$.

We then used OFE (resp., DEP) to learn the parameters for the NaïveBayes structure (Figure 1(a)) from a data sample, then used the resulting BN to classify additional data. As the structure was correct for this P_0 distribution, both OFE and DEP did quite well, efficiently converging to the optimal classification error.

We then considered learning the CPtables for this NaïveBayes structure, but for distributions that were *not* consistent with this structure. We therefore formed the m^{th} distribution P_m by asserting that $E_1 \equiv E_2 \equiv \ldots \equiv E_m$ (*i.e.*, $P(\,E_i \,|\, E_1\,) = 1.0$, $P(\,E_i \,|\, \neg E_1\,) = 0.0$ for each $i = 1..m$) in addition to Equation 8. Hence, P_0 corresponds to the $m = 0$ case. For $m > 0$, however, the m-th distribution cannot be modeled as a NaïveBayes structure, but could be modeled using that structure augmented with $m-1$ links, connecting E_{i-1} to E_i for each $i = 2..m$.

Figure 10(a) shows the results, for $k = 5$, based on 400 instances. We see that, as predicted, DEP can produce reasonably accurate CPtables here, even for increasingly wrong structures. However, OFE does progressively worse.

(The "blip" at $m = 1$ is due to the quantization of selecting a classification value; we see a strictly monotonic relation if we instead consider "mean squared error" (Equation 7); see Figure 10(b).

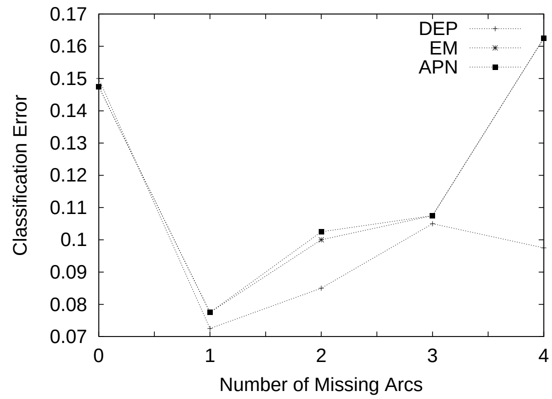


Figure 11: Comparing DEP to OFE, on Increasingly Incorrect Structures (NaïveBayes) using In-Complete Data; using Classification Error

Notice that the DEP error actually becomes processively *smaller* as the structure become worse, *i.e.*, as we increase the number of additional links in the true model. This is because, for DEP, each additional link (in the true model) makes the task easier, as there are fewer degrees of freedom. Of course this does not help the maximal likelihood approach OFE, because it is forced to consider only the immediate connections provided by the links.)

4.5 "Correctness of Structure" Study... Incomplete Data

We next degraded this training data by randomly removing the value of each attribute, within each instance, with probability 0.5. We then compared DEP with the standard systems APN and EM. These results appear in Figure 11. Here, we again see that DEP is more accurate, in each case.

5 Discussion

Executive Summary: Our empirical studies on the UCI datasets suggest, when given complete training data,

$$\boxed{\begin{array}{lclcc} \textbf{TAN+}\textsc{dep} & > & \text{TAN+}\textsc{ofe} & & \\ & \succeq & \textbf{NB+}\textsc{dep} & \succeq & \text{C4.5} \\ & & & > & \text{SNB+}\textsc{ofe} \\ & & & > & \text{NB+}\textsc{ofe} \end{array}} \tag{9}$$

and when dealing with incomplete data,

$$\boxed{\begin{array}{ccc} \textbf{NB+}\textsc{dep} & > & \left\{ \begin{array}{c} \text{NB+}\textsc{apn} \\ \text{NB+}\textsc{em} \end{array} \right\} \\ \left\{ \begin{array}{c} \textbf{TAN+}\textsc{dep} \\ \approx \\ \text{TAN+}\textsc{apn} \end{array} \right\} & \succeq & \text{TAN+}\textsc{em} \end{array}} \tag{10}$$

We use "$>$" to indicate a statistical significance at the $p < 0.05$ level or better; and "$\succeq$" to indicate plausibly better, at the level of $p < 0.2$ or better.

Why DEP **Works Well:** In a nutshell, DEP worked effectively in many situations; and it was especially advantageous (*i.e.*, typically better than the alternative ways to instantiate parameters) whenever the BN-structure was *incorrect* — *i.e.*, whenever it is not an I-map of the underlying distribution by incorrectly claiming that two dependent variables are independent [Pea88]. This is a very common situation, as many BN-learners will produce incorrect structures, either because they are conservative in adding new arcs (to avoid overfitting the data [Hec98, VG00]), or because they are considering only a restricted class of structures (*e.g.*, NaïveBayes [DH73], poly-tree [CL68, Pea88], Tree-Augmented Network [FGG97], etc.) which is not guaranteed to contain the correct structure.

To understand why a bad structure is problematic for OFE, recall that OFE is guaranteed to produce the parameter values that have the optimal likelihood value *for the structure* G, given the data S. However, if the structure G is incorrect, even the optimal-likelihood-for-G parameters might yield a fairly poor model of the true tuple distribution, which means it might return incorrect values for queries. By contrast, the DEP algorithm is not as constrained by the specific structure, and so may be able to produce parameters that yield fairly accurate answers, even if the structure is sub-optimal. (See the standard comparison between discriminative versus generative training, overviewed in Section 6 below.)

Computational Efficiency: Our current DEP implementation — in unoptimized JAVA code — was relatively slow. In general, it required a handful of iterations to converge for the small datasets, and dozens for the larger ones. APN and EM typically used slightly more iterations. DEP's time per iteration varied, from around 0.5 seconds per iteration for the smaller datasets through a few minutes for larger datasets Again, this is roughly comparable to the performance of the incomplete data algorithms, APN and EM.

These times are, of course, considerable more than required by OFE, which is arguably the most efficient possible algorithm. We are currently investigating whether there could be a more efficient algorithm for our task (computing parameters that optimize conditional likelihood) in the complete data case.

Tradeoff: Our results, in general, suggest an interesting tradeoff: Most BN-learners spend most of their time learning a near-optimal structure [CGH94], then use a simple algorithm (OFE) to fill in the CPtables. When the goal is classification accuracy, our empiricial studies suggest instead quickly producing trivial structures — such as NaïveBayes — then spending time learning good parameters, using DEP.

6 Related Results

There are a number of researchers providing techniques, and insights, related to learning belief nets. Much of their work focuses on learning the best *structure*, either for a general belief net, or within the context of some specific class of structures (*e.g.*, TAN-structures, or selective NaïveBayes); see [Hec98, Bun96] for extensive tutorials. By contrast, this paper suggests a way to learn the *parameters* for a given structure.

Most of those structure-learning systems also learn the parameters. Essentially all use the OFE algorithm here (Equation 6). This is well motivated in the generative situation, as these parameter values do optimize the likelihood of the data [CH92].

As noted earlier, our goal is different, as we are seeking the optimal *classifier* — *i.e.*, *discriminant learning*. While a perfect model of the underlying distribution would also be the optimal classifier, the converse is not true; *i.e.*, we are happy with parameters that yield a good classifier, even if those parameters do not reflect the true underlying distribution. That is, the eventual performance system will be expected to address a certain range of questions — *e.g.*, about the probability of cancer given gender, smoking habits, etc. We consider our learner good if it produces parameters that provide appropriate answers to these questions, even if the overall distribution would return completely wrong answers to other (unasked) questions, *e.g.*, about the conditional probability of smoking given gender, etc.

There have been many other systems that also considered discriminant learning of belief nets [KMST99, JMJ00, FGG97, CG99]. These systems, however, focused on structure learning; and usually used OFE to instantiate the resulting parameters.

Our results also relate closely to the work on *discriminant learning of Hidden Markov Models (HMMs)* [SMK+97, CJL92]. In particular, much of that work uses "Generatized Probabilistic Descent", which resembles our DEP system by descending along the derivative of the parameters, to maximize the conditional likelihood of the hypothesis (which typically are words) given the observations — which they call "Maximimum Mutual Information" criterion.

This relates directly to the large literature on *discriminant learning* in general; see [CS89, Jor95, Rip96]. One standard model is Linear Discriminant Analysis (LDA), which typically assumes $P(\mathbf{E} \mid C=c)$ is multivariate normal (*i.e.*, $P(\mathbf{E} \mid C=c) \sim \mathcal{N}(\mu_c, \Sigma)$ where each μ_c mean is dependent on the class $C=c$, and the covariance matrix is the same for all classes). The LDA system then estimates the relevant $\{\mu_c, \Sigma, \hat{P}(C=c)\}$ parameters from a body of data, seeking the ones that maximize the likelihood of the data relevant to that parameter. Given this model, it can then use Bayes Rule to compute the posterior distribution of $P(C \mid \mathbf{E}=\mathbf{e}')$ given new evidence $\mathbf{E}=\mathbf{e}'$.

Notice this approach is, in essense, generative, in that it deals with the entire $\langle C; \mathbf{E} \rangle$ joint distribution. By contrast, Multiple Logistic Regression (MLR) estimates the parameters explicitly associated with the posterior distribution, of the form

$$P(C=c \mid \mathbf{E}=\mathbf{e}) \quad = \quad \frac{\exp(\alpha_c + \beta_c \cdot \mathbf{e})}{\sum_j \exp(\alpha_j + \beta_j \cdot \mathbf{e})}$$

seeking the $\{\alpha_j, \beta_j\}$ parameters that maximize the conditional likelihood $P(c \mid \mathbf{e})$. Note this form corresponds to the multivariate conditions used by LDA; indeed, this form is appropriate whenever $P(\mathbf{E} \mid C)$ is in the exponential family.

We can view LDA as being generative (aka "causal" or "class-conditional" [Jor95], or "sampling" [Daw76]), as it is attempting to fit parameters for the entire joint distribution, while MLR is discriminant (aka "diagnostic", "predictive" [Jor95]), as it focuses only on the conditional probabilities. We can therefore identify LDA with (generative) OFE, and MLR with (discriminant) DEP.

Our results echo the common wisdom obtained by these prior analyses of discriminant systems. In particular, (1) accuracy: discriminant training typically produces more accurate classifiers than generative training (see the comparative studies throughout Section 4); (2) robustness: typically discriminant is more robust against incorrect models than generative (see Section 4.4 and 4.5); (3) efficiency: typically generative is more efficient than discriminant (compare the efficient OFE with the NP-hard DEP). Due to the final point, many discriminant learners initialize their parameters based on generative (read "maximum-likelihood") estimates, especially as the latter are often "plug-in parameters" [Rip96]; our OFE-DEP algorithm incorporates this idea as well.

The work reported in this paper has significant differences, of course. First, we are dealing with a different underlying model, based on *discrete* variables (rather than normal ones), in the context of a *specified belief net structure*, which corresponds to a given set of independency claims. (While our empirical evidence focuses on NaïveBayes and TAN structures, our algorithm of course works for arbitrary structures.) We also describe the inherent computational complexity of this task, produce algorithms specific to our task, and provide empirical studies to demonstrate that our algorithms works effectively, given either complete or incomplete training data.

Finally, our companion paper [GGS97] also considers learning the parameters of a given structure towards optimizing performance on a distribution of queries. Our results here differ, as we are considering a different learning model: [GGS97] tries to minimize the squared-error score (a variant of Equation 7) which is based on two different types of samples — one with tuples, to estimate $P(\,C \mid \mathbf{E}\,)$, and the other with queries, to estimate the probability of seeing each "What is $P(\,C \mid \mathbf{E}\!=\!\mathbf{e}\,)$?" query. By contrast, the current paper tries to minimize classification error (Equation 1) by seeking the optimal "conditional likelihood" score (Equation 3), wrt a single sample of labeled instances. Also, of course, our current paper includes new theoretical results, a different algorithm, and completely new empirical data.

7 Conclusions

7.1 Future Work

1. This paper investigtes the challenges of filling in the CPtables of a given BN-structure. While this is an important subtask, a general learner should be able to learn that structure as well — perhaps using *conditional likelihood* as the selection criterion; see [KMST99, JMJ00]. We plan to investigate ways to synthesize these approaches.

2. While the LCL-optimization task is NP-hard in general (Theorem 1), there may be useful special cases where this task is easy. We are currently investigating whether the learning task is easy when the data is complete.

7.2 Contributions

Most BN-learners seek the network that maximizes likelihood of the data; this is appropriate when the learner's goal is obtaining an apt model of the underlying distribution. In many cases, however, the learner is trying to produce an accurate classifier; here it makes sense to use "discriminant" learning techniques — typically by seeking the parameters that maximize the *conditional* likelihood.

This paper focuses on the specific task of finding the parameters for a fixed belief net structure, over discrete variables. After showing that this task is intractable in general, we provide an effective gradient-descent learning algorithm DEP and demonstrate empirically that DEP works well in practice over a variety of situations (*e.g.*, both complete and incomplete data) — in many cases, performing better than the algorithms developed for the more familiar "maximize likelihood" context; see Equations 9 and 10. We also illustrate that our DEP works especially well in the common cases where the structure is incorrect.

Our results are consistent with the standard knowledge about discriminant vs generative learning. We contribute to this knowledge by provide a concrete demonstration in our specific context: estimation of discrete parameters for a fixed structure, wrt a 0/1 loss function, etc. In addition, while the idea of using gradient descent here is not original, some aspects of our specific algorithm may provide insights to future researchers considering this approach — in particular, the various tricks we use to reduce the computation (*e.g.*, effectively ignoring the d-seperated parameters), and our use of the "softmax" encoding to satisfy the *constrained* optimization task.

Table 1: Description of data sets used in the experiments; see [FGG97]

	Dataset	# Attributes	# Classes	# Instances	
				Train	Test
1	australian	14	2	690	CV-5
2	breast	10	2	683	CV-5
3	chess	36	2	2130	1066
4	cleve	13	2	296	CV-5
5	corral	6	2	128	CV-5
6	crx	15	2	653	CV-5
7	diabetes	8	2	768	CV-5
8	flare	10	2	1066	CV-5
9	german	20	2	1000	CV-5
10	glass	9	7	214	CV-5
11	glass2	9	2	163	CV-5
12	heart	13	2	270	CV-5
13	hepatitis	19	2	80	CV-5
14	iris	4	3	150	CV-5
15	letter	16	26	15000	5000
16	lymphography	18	4	148	CV-5
17	mofn-3-7-10	10	2	300	1024
18	pima	8	2	768	CV-5
19	satimage	36	6	4435	2000
20	segment	19	7	1540	770
21	shuttle-small	9	7	3866	1934
22	soybean-large	35	19	562	CV-5
23	vehicle	18	4	846	CV-5
24	vote	16	2	435	CV-5
25	waveform-21	21	3	300	4700

Acknowledgements

We thank Tom Dietterich, Charles Elkan, Adam Grove, Peter Hooper, Daphne Koller, Dale Schuurmans, and Lyle Ungar for their many helpful suggestions. We also thank Fabio Cozman for making his JavaBayes code available, which is the foundation of our implementation. RG was partially funded by NSERC, and by Siemens Corporate Research; and WZ was partially funded by NSERC and by Syncrude.

References

[Bis98] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford, 1998.

[BKRK97] John Binder, Daphne Koller, Stuart Russell, and Keiji Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29:213–244, 1997.

[BM00] C. Blake and C. J. Merz. UCI repository of machine learning databases. Technical report, Dept. Info. & Comp. Sci., Univ. Calif. at Irvine, 2000. http://www.ics.uci.edu/~mlearn/MLRepository.html.

Table 2: Empirical Results (Accuracy) of 6 approaches to Learning Classifiers, over 25 datasets, for complete data (SNB+OFE, C4.5 data from [FGG97, Table 3])

	Data set	NB+OFE	TAN+OFE	NB+DEP	TAN+DEP	SNB+OFE	C4.5
1	AUSTRALIAN	86.81±0.84	85.07±1.09	84.93±1.06	85.07±1.09	86.67±1.81	85.65±1.82
2	BREAST	96.55±0.83	96.55±0.58	95.54±0.48	96.12±0.37	96.19±0.63	94.73±0.59
3	CHESS	87.34±1.02	92.40±0.81	95.40±0.64	97.09±0.51	94.28±0.71	99.53±0.21
4	CLEVE	82.33±3.27	82.00±3.78	82.33±3.89	81.33±4.33	78.06±2.41	73.31±0.63
5	CORRAL	86.40±2.99	99.20±0.80	90.40±1.60	100.00±0.00	83.57±3.15	97.69±2.31
6	CRX	86.09±1.49	85.07±1.14	84.64±1.15	85.07±1.14	85.92±1.08	86.22±0.58
7	DIABETES	75.56±1.32	75.82±1.17	75.69±1.26	75.95±1.25	76.04±0.83	76.04±0.85
8	FLARE	79.62±1.17	83.29±1.75	82.72±2.22	82.35±2.14	83.40±1.67	82.55±1.75
9	GERMAN	73.50±1.45	73.60±0.94	74.00±1.28	73.60±0.94	73.70±2.02	72.20±1.23
10	GLASS	41.90±2.45	41.90±2.45	41.90±2.45	41.90±2.45	71.98±2.15	69.62±1.95
11	GLASS2	76.25±3.22	75.63±3.03	77.50±3.34	76.25±3.22	79.17±1.71	76.67±1.63
12	HEART	78.89±3.91	81.11±3.38	79.26±3.81	80.00±3.68	81.85±2.83	81.11±3.77
13	HEPATITIS	81.94±1.29	85.16±2.41	85.16±0.79	85.16±1.64	90.00±4.24	86.25±4.15
14	IRIS	95.33±0.82	94.67±0.82	95.33±0.82	95.33±0.82	94.00±1.25	94.00±1.25
15	LETTER	72.72±0.63	84.52±0.51	83.54±0.52	88.90±0.44	75.36±0.61	77.70±0.59
16	LYMPHOGRAPHY	82.76±3.27	78.62±1.29	83.45±2.53	79.31±1.09	77.72±2.46	77.03±1.21
17	MOFN-3-7-10	86.72±1.06	90.72±0.91	100.00±0.00	100.00±0.00	87.50±1.03	85.55±1.10
18	PIMA	73.46±1.12	75.42±0.84	75.42±1.00	75.69±0.76	74.86±2.61	75.13±1.52
19	SATIMAGE	81.70±0.86	87.85±0.73	85.50±0.79	88.60±0.71	82.05±0.86	83.15±0.84
20	SEGMENT	85.32±1.28	89.35±1.11	89.74±1.09	89.74±1.09	93.25±0.90	93.64±0.88
21	SHUTTLE-SMALL	98.86±0.24	99.38±0.18	99.28±0.19	99.38±0.18	99.28±0.19	99.17±0.21
22	SOYBEAN-LARGE	92.65±0.33	91.47±1.27	92.65±0.66	92.65±1.54	92.89±1.01	92.00±1.11
23	VEHICLE	56.80±1.46	61.07±1.10	62.72±1.62	64.97±0.68	61.36±2.33	69.74±1.52
24	VOTE	90.34±2.32	94.25±2.06	96.09±1.39	95.40±1.50	94.71±0.59	95.63±0.43
25	WAVEFORM-21	76.55±0.62	76.30±0.62	78.45±0.60	76.74±0.62	76.53±0.62	74.70±0.63

Table 3: Empirical Results (Accuracy) of 6 approaches to Learning Classifiers, from incomplete data

	Data set	NB+EM	TAN+EM	NB+APN	TAN+APN	NB+DEP	TAN+DEP
1	AUSTRALIAN	83.62±1.69	82.90±1.48	83.62±1.60	83.62±1.20	83.33±1.65	82.32±0.96
2	BREAST	83.60±6.43	83.74±6.49	83.60±6.43	83.60±6.43	83.31±6.29	83.74±6.49
3	CHESS	81.89±1.18	84.80±1.10	82.08±1.17	84.62±1.11	84.62±1.11	87.24±1.02
4	CLEVE	81.00±3.40	80.67±4.07	82.00±3.70	81.67±4.74	80.33±2.07	80.33±3.89
5	CORRAL	78.40±2.71	85.60±2.40	78.40±2.71	88.80±3.44	78.40±2.40	86.40±2.71
6	CRX	82.75±0.77	82.32±0.75	82.75±0.96	81.30±0.93	83.04±0.88	82.32±0.75
7	DIABETES	67.19±2.90	67.06±3.11	67.84±2.84	67.06±3.11	67.84±2.84	67.06±2.99
8	FLARE	80.28±1.58	83.19±2.03	80.00±1.53	81.50±1.64	82.72±2.10	82.54±1.95
9	GERMAN	73.10±0.97	70.50±1.08	73.40±1.04	70.00±0.92	72.80±1.22	70.50±1.08
10	GLASS	30.48±3.32	30.48±3.32	30.48±3.32	30.48±3.32	30.48±3.32	30.48±3.32
11	GLASS2	55.63±1.53	55.63±1.53	55.63±1.53	55.63±1.53	55.63±1.53	55.63±1.53
12	HEART	72.96±4.04	72.22±3.56	73.33±3.73	72.22±3.56	73.33±3.91	73.70±3.99
13	HEPATITIS	82.58±0.79	80.00±1.88	82.58±0.79	80.65±1.02	80.65±2.04	79.35±1.64
14	IRIS	48.67±10.98	48.67±10.98	49.33±10.61	48.67±10.36	57.33±11.22	57.33±11.22
15	LETTER	3.70±0.27	3.70±0.27	3.70±0.27	3.70±0.27	3.70±0.27	3.70±0.27
16	LYMPHOGRAPHY	82.07±3.84	75.86±1.09	82.07±2.97	75.17±3.68	82.07±4.14	75.86±1.09
17	MOFN-3-7-10	84.57±1.13	86.62±1.06	84.57±1.13	85.74±1.09	88.28±1.01	87.30±1.04
18	PIMA	70.46±0.96	70.59±0.92	70.72±1.04	70.85±1.00	70.72±1.04	70.59±0.92
19	SATIMAGE	67.55±1.05	70.60±1.02	67.20±1.05	71.65±1.01	70.65±1.02	71.25±1.01
20	SEGMENT	72.60±1.61	72.86±1.60	72.34±1.61	72.73±1.60	73.12±1.60	72.60±1.61
21	SHUTTLE-SMALL	78.90±0.93	78.90±0.93	78.90±0.93	78.90±0.93	78.90±0.93	78.90±0.93
22	SOYBEAN-LARGE	88.82±0.91	76.47±1.12	89.56±0.85	83.24±1.34	89.56±1.26	82.06±1.72
23	VEHICLE	43.55±2.77	49.47±4.25	43.20±2.92	49.11±4.51	47.34±4.20	48.52±4.61
24	VOTE	89.66±2.21	94.94±0.86	89.66±2.21	95.17±0.92	92.64±1.34	94.25±1.21
25	WAVEFORM-21	70.70±0.66	70.15±0.67	70.85±0.66	69.68±0.67	70.04±0.67	70.15±0.67

[Bun96] Wray Buntine. A guide to the literature on learning probabilistic networks from data. *IEEE Transactions on Knowledge and Data Engineering*, 1996.

[CG99] Jie Cheng and Russell Greiner. Comparing bayesian network classifiers. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI'99)*, pages 101–107. Morgan Kaufmann Publishers, August 1999.

[CGH94] David M. Chickering, Dan Geiger, and David Heckerman. Learning bayesian networks is NP-hard. Technical Report MSR-TR-94-17, Microsoft Research, November 1994.

[CH92] G. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning Journal*, 9:309–347, 1992.

[CJL92] W. Chou, B. Juang, and C. Lee. Segmental GPD training of HMM based speech recognizer. In *ICASSP*, volume 1, pages 473–476, 1992.

[CL68] C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Information Theory*, pages 462–467, 1968.

[Coo90] G.F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2–3):393–405, 1990.

[CS89] D. R. Cox and E. J. Snell. *Analysis of Binary Data.* Chapman & Hall, London, 1989.

[CY99] G. Cooper and C. Yoo. Causal discovery from a mixture of experimental and observational data. In *UAI99*, 1999.

[Daw76] A. P. Dawid. Properties of diagnostic data distributions. *Biometrics*, 32:647–658, 1976.

[DH73] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis.* Wiley, New York, 1973.

[DP96] P. Domingo and M. Pazzani. Beyond independence: conditions for the optimality of the simple bayesian classier. In *Proc. 13th International Conference on Machine Learning*, 1996.

[FGG97] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine Learning Journal*, 29:131–163, 1997.

[FI93] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the Thirteeth International Joint Conference on Artificial Intelligence*, pages 1022–1027, San Francisco, CA, 1993. Morgan Kaufmann.

[GGS97] Russell Greiner, Adam Grove, and Dale Schuurmans. Learning Bayesian nets that perform well. In *Uncertainty in Artificial Intelligence*, 1997.

[Hec98] David E. Heckerman. A tutorial on learning with Bayesian networks. In M. I. Jordan, editor, *Learning in Graphical Models*, 1998.

[JMJ00] T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *NIPS2000*, 2000.

[Jor95] M. Jordan. Why the logistic function? a tutorial discussion on probabilities and neural networks, 1995.

[KJ97] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2), 1997.

[KMST99] Petri Kontkanen, Petri Myllymäki, Tomi Silander, and Henry Tirri. On supervised selection of bayesian networks. In *UAI99*, pages 334–342, 1999.

[Koh95] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1137–1143, San Francisco, CA, 1995. Morgan Kaufmann.

[KS96] Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *Proc. 13th International Conference on Machine Learning*, pages 284–292. Morgan Kaufmann, 1996.

[LIT92] Pat Langley, Wayne Iba, and Kevin Thompson. An analysis of bayesian classifiers. In William Swartout, editor, *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 223–228, San Jose, CA, July 1992. MIT Press.

[LR87] J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data.* Wiley, New York, 1987.

[Mit97] Tom M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, 1988.

[Qui92] J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann Publishers, San Mateo, 1992.

[Rip96] B. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, Cambridge, UK, 1996.

[SMK+97] R. Schlüter, W. Macherey, S. Kanthak, H. Ney, and L. Welling. Comparison of optimization methods for discriminative training criteria. In *Proc. EUROSPEECH'97*, pages 15–18, 1997.

[Thi95] B. Thiesson. Accelerated quantification of bayesian networks with incomplete data. In *Proceedings of First International Conference on Knowledge Discovery and Data Mining*, pages 306–311, 1995.

[Vap98] V. Vapnik. *Statistical Learning Theory.* Springer, 1998.

[VG00] Tim Van Allen and Russell Greiner. Model selection criteria for learning belief nets: An empirical comparison. In *ICML'00*, pages 1047–1054, 2000.

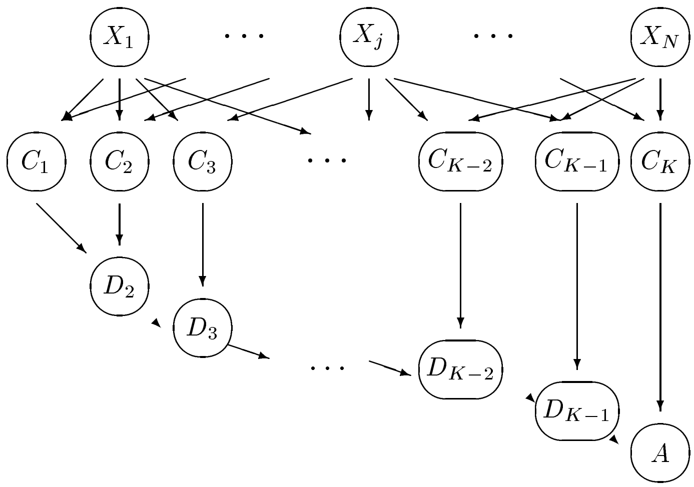


Figure 12: Belief Net structure for any SAT problem [Coo90]

A Proofs

Proof of Theorem 1: We reduce 3SAT to our task, using a construction similar to the one in [Coo90]: Given any 3-CNF formula $\varphi \equiv \bigwedge C_i$, where each $C_i \equiv \bigvee \pm X_{ij}$, we construct the network shown in Figure 12, with one node for each variable X_i and one for each clause C_j, with an arc from X_i to C_j whenever C_j involves X_i — *e.g.*, if $C_1 = x_1 \vee \bar{x}_2 \vee x_3$ and $C_2 = \bar{x}_1 \vee \bar{x}_3 \vee x_4$, then there are links to C_1 to X_1, X_2 and X_3, and from C_2 from X_1, X_3 and X_4. In addition, we include $K-1$ other boolean nodes, $\{D_2, \ldots, D_{K-1}, A\}$, where D_j is the child of D_{j-1} and C_j, where D_1 is identified with C_1, and A is used for D_K.

Here, we intend each C_i to be true if the assignment to the associated variables X_{i1}, X_{i2}, X_{i3} satisfies C_i; and A corresponds is the conjunction of those variables. We do this using (all-but-the-final) instances in Figure 13. There is one such instance for each clause, with exactly the assignment (of the 3 relevant variables) that falsifies this clause. Hence, the first line corresponds to $C_1 \equiv x_1 \vee \bar{x}_2 \vee X_3$. One complication here is that the "label" of each instance must always be the single variable A.

We now prove, in particular, that

> There is a set of parameters for the structure in Figure 12, producing 0 $\widehat{\mathrm{LCL}}^{()}(\ )$-score over the queries in Figure 13,
> *iff* there is a satisfying assignment for the associated φ formula.

$\Leftarrow$: Just set each C_i to be the disjunction of the associated X_{i1}, X_{i2}, X_{i3} variables (its parents), with the appropriate sense. Eg, using $C_1 \equiv x_1 \vee \bar{x}_2 \vee X_3$, then C_1's CPtable would have

X_1	X_2	X_3	$P(+c_1 \mid X_1, X_2, X_3)$
0	0	0	1.0
0	0	1	1.0
0	1	0	1.0
0	1	1	0.0
1	0	0	1.0
1	0	1	1.0
1	1	0	1.0
1	1	1	1.0

. Similarly set the CPtables for the D_j to correspond to the

X_1	X_2	X_3	X_4	$\cdots$	X_n	A
0	1	0				0
0		0	1			0
	$\vdots$					$\vdots$
0		1		1		0
						1

Figure 13: Queries used in Proof of Theorem 1

conjunction of its 2 parents $D_j = D_{j-1} \wedge C_j$; *e.g.*,

D_4	C_5	$P(+d_5 \mid D_4, C_5)$
0	0	0.0
0	1	0.0
1	0	0.0
1	1	1.0

.

Finally, set X_i to correspond to the satisfying assignment; *i.e.*, if $X_1 = 1$, then $\boxed{\frac{P(+x_1)}{1.0}}$; and if *i.e.*, if $X_4 = 0$, then $\boxed{\frac{P(+x_4)}{0.0}}$. Note that these CPtable values satify all $k+1$ of the labeled instances.

$\Leftarrow$: Here, we assume there is no satisfying assignment. Towards a contradiction, we can assume that there is a 0-LCL set of CPtable entries. This means, in particular, that $P(+a \mid x_{i1}, x_{i2}, x_{i3}) = 0$, where x_{i1}, x_{i2}, x_{i3} correspond to the assignment that violates the ith constraint. (*E.g.*, for $C_1 \equiv X_1 \vee \neg X_2 \vee X_3$, this would be $X_1 = 0$, $X_2 = 1$, $X_3 = 0$.)

Now consider the final labled instance, $P(a)$. As there is no satisfying assignment, we know that each assignment $\mathbf{x}$ violates at least one constraint. For notation, let $\gamma^{\mathbf{x}}$ refer to one of these violations (say the one with the smallest index). So if $\mathbf{x} = \langle 0, 1, 0, \ldots \rangle$, then $\gamma^{\langle 0,1,0,\ldots\rangle} = \langle X_1 = 0,\ X_2 = 1,\ X_3 = 0 \rangle$ corresponds to the violation of the first constraint C_1. We also let $\beta^{\mathbf{x}}$ refer to the rest of the assignment.

Now observe

$$\begin{aligned} P(+a) &= \textstyle\sum_{\mathbf{x}} P(+a,\ \mathbf{x}) \\ &= \textstyle\sum_{\mathbf{x}} P(+a \mid \gamma^{\mathbf{x}}) \cdot P(\gamma^{\mathbf{x}}) \cdot P(\beta^{\mathbf{x}} \mid +a,\ \gamma^{\mathbf{x}}) \\ &= \textstyle\sum_{\mathbf{x}} \quad 0 \quad \cdot P(\gamma^{\mathbf{x}}) \cdot P(\beta^{\mathbf{x}} \mid +a,\ \gamma^{\mathbf{x}}) \quad = \quad 0 \end{aligned}$$

which shows that the final instance will be mislabeled. This proves that there can be no set of CPtable values that produce 0 LCL-score when there are no satisfying assignments. ∎

Proof of Proposition 2: Below, we will use $P(\chi)$ to refer to $P_B(\chi)$, the value the belief net B will assign to the χ event. In general, for any assignment Z,

$$P(Z) \quad = \quad \sum_{\mathbf{f}'} \sum_{r'} P(Z \mid R{=}r', \mathbf{F}{=}\mathbf{f}')\ P(R{=}r' \mid \mathbf{F}{=}\mathbf{f}')\ P(\mathbf{F}{=}\mathbf{f}') \tag{11}$$

As we assume the different CPtable rows are independent, and $\mathbf{F}$ are the parents of R, this means

$$\frac{\partial\, P(B \mid Z)}{\partial\, \beta_{r|\mathbf{f}}} \quad = \quad \sum_{r'} P(Z \mid r', \mathbf{f})\ \frac{\partial\, P(r' \mid \mathbf{f})}{\partial\, \beta_{r|\mathbf{f}}} P(\mathbf{f})$$

Recalling $\theta_{r|\mathbf{f}} = P(r \mid \mathbf{f}) = e^{\beta_{r|\mathbf{f}}} / \sum_{r'} e^{\beta_{r'|\mathbf{f}}}$, observe that $\frac{\partial\, P(r \mid \mathbf{f})}{\partial\, \beta_{r|\mathbf{f}}} = \theta_{r|\mathbf{f}}(1 - \theta_{r|\mathbf{f}})$, and when $r \neq r'$, $\frac{\partial\, P(r' \mid \mathbf{f})}{\partial\, \beta_{r|\mathbf{f}}} = -\theta_{r|\mathbf{f}}\, \theta_{r'|\mathbf{f}}$. This means $\frac{\partial\, P(Z)}{\partial\, \beta_{r|\mathbf{f}}} = P(Z, r, \mathbf{f}) - \theta_{r|\mathbf{f}} P(Z, \mathbf{f})$.

Hence, as $\ln P(\,c\,|\,\mathbf{e}\,) \;=\; \ln P(\,c, \mathbf{e}\,) \;-\; \ln P(\,\mathbf{e}\,)$,

$$\begin{aligned}
\frac{\partial\; \ln P(\,c\,|\,\mathbf{e}\,)}{\partial\; \beta_{r|\mathbf{f}}} &= \frac{\partial\; \ln P(\,c, \mathbf{e}\,)}{\partial\; \beta_{r|\mathbf{f}}} - \frac{\partial\; \ln P(\,\mathbf{e}\,)}{\partial\; \beta_{r|\mathbf{f}}} \\
&= \frac{1}{P(\,c, \mathbf{e}\,)} \frac{\partial\; P(\,c, \mathbf{e}\,)}{\partial\; \beta_{r|\mathbf{f}}} - \frac{1}{P(\,\mathbf{e}\,)} \frac{\partial\; P(\,\mathbf{e}\,)}{\partial\; \beta_{r|\mathbf{f}}} \\
&= \frac{1}{P(\,c, \mathbf{e}\,)} [P(\,c, \mathbf{e}, r, \mathbf{f}\,) - \theta_{r|\mathbf{f}} P(\,c, \mathbf{e}, \mathbf{f}\,)] \;-\; \frac{1}{P(\,\mathbf{e}\,)} [P(\,\mathbf{e}, r, \mathbf{f}\,) - \theta_{r|\mathbf{f}} P(\,\mathbf{e}, \mathbf{f}\,)] \\
&= [P(\,r, \mathbf{f}\,|\,c,\,\mathbf{e}\,) - P(\,\mathbf{f}\,|\,c,\,\mathbf{e}\,)] \;-\; \theta_{r|\mathbf{f}}\,[P(\,\mathbf{f}\,|\,c,\,\mathbf{e}\,) - P(\,\mathbf{f}\,|\,\mathbf{e}\,)]
\end{aligned}$$

∎

Proof of Proposition 3: For a given NaïveBayes structure with root C and attributes A_i, and parameter values Θ, the log likelihood function for data $D = \{\langle a_1^i, \ldots, a_n^i \rangle, c^i\}_{i=1..M}$ with M iid cases is

$$\begin{aligned}
LL_\Theta(D) &= \sum_{i=1}^{M} \ln P_\Theta(\,a_1^i, \ldots, a_n^i, c^i\,) \\
&= \sum_{i=1}^{M} \ln \left(P_\Theta(\,c^i\,) \quad \times \quad \prod_j P_\Theta(\,a_j^i\,|\,c^i\,) \right) \\
&= \sum_{i=1}^{M} \left(\ln P_\Theta(\,c^i\,) \quad + \quad \sum_j \ln P_\Theta(\,a_j^i\,|\,c^i\,) \right)
\end{aligned}$$

Now notice each of the terms can be maximized independently. If any a_j^i's are missing, we can simply omit these these in the summation because they will not contribute to the likelihood score; *i.e.*, $P_\Theta(\,C = c, A = ?\,)$ is just calculated as $P_\Theta(\,C = c\,)$ This means

$$LL_\Theta(D) \quad = \quad \sum_{i=1}^{M} \ln P_\Theta(\,c^i\,) \quad + \quad \sum_j \sum_{i=1..M, a_j^i \neq ?} \ln P_\Theta(\,a_j^i\,|\,c^i\,))$$

To maximize this sum, we need only maximize each individual term, which corresponds to setting each parameters to its observed frequency $\theta_{A_i=x|C=y} = \frac{\#(A_i=x, C=y)}{\#(C=y)}$, ignoring (for A_i) the records where A_i was not mentioned. ∎