Structural Extension to Logistic Regression:

Discriminative Parameter Learning of Belief Net Classifiers

Russell Greiner (greiner@cs.ualberta.ca) *
Dept of Computing Science, University of Alberta, Edmonton, AB T6G 2H1 Canada

Xiaoyuan Su (xiaoyuan@cs.ualberta.ca)
Electrical & Computer Engineering, University of Miami, Coral Gables, FL 33124, USA

Bin Shen (bshen@cs.ualberta.ca)
Dept of Computing Science, University of Alberta, Edmonton, AB T6G 2H1 Canada

Wei Zhou (w2zhou@math.uwaterloo.ca)
Dept of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada

Abstract. Bayesian belief nets (BNs) are often used for classification tasks — typically to return the most likely class label for each specified instance. Many BN-learners, however, attempt to find the BN that maximizes a different objective function — viz., likelihood, rather than classification accuracy — typically by first learning an appropriate graphical structure, then finding the parameters for that structure that maximize the likelihood of the data. As these parameters may not maximize the classification accuracy, "discriminative parameter learners" follow the alternative approach of seeking the parameters that maximize *conditional likelihood* (CL), over the distribution of instances the BN will have to classify. This paper first formally specifies this task, shows how it extends standard logistic regression, and analyzes its inherent sample and computational complexity. We then present a general algorithm for this task, ELR, that applies to arbitrary BN structures and that works effectively even when given incomplete training data. Unfortunately, ELR is not guaranteed to find the parameters that optimize conditional likelihood; moreover, even the optimal-CL parameters need not have minimal classification error. This paper therefore presents empirical evidence that ELR produces effective classifiers, often superior to the ones produced by the standard "generative" algorithms, especially in common situations where the given BN-structure is incorrect.

Keywords: (Bayesian) belief nets, Logistic regression, Classification, PAC-learning, Computational/sample complexity

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 specified by a set of attributes. An increasing number of projects are using "(Bayesian) belief nets" (*BN*) to represent the underlying distribution, and hence the stochastic mapping from evidence to response.

---

* This e-mail address is available for all problems and questions. This paper extends the earlier results that appear in (SSG$^+$03) and (GZ02).

When this distribution is not known *a priori*, we can try to *learn* the model. Our goal is an *accurate* BN — *i.e.*, one that returns *the correct answer as often as possible*. While a perfect model of the distribution will perform optimally for any possible query, learners with limited training data are unlikely to produce such a model; moreover, optimality may be impossible for learners constrained to a restricted range of possible distributions that excludes the correct one (*e.g.*, when only considering parameterizations of a given *BN*-structure).

Here, it makes sense to find the parameters that do well with respect to the queries posed. This "discriminative learning" task differs from the "generative learning" that is used to learn an overall model of the distribution (Rip96). Following standard practice, our discriminative learner will seek the parameters that maximize the *log conditional likelihood* (LCL) over the data, rather than simple likehood — that is, given the data $S = \{\langle c_i, \mathbf{e}_i\rangle\}$ (each class label $C = c_i$ associated with evidence $\mathbf{E} = \mathbf{e}_i$), a discriminative learner will try to find parameters Θ that maximize

$$\widehat{\mathrm{LCL}}^{(S)}(\Theta) \quad = \quad \frac{1}{|S|} \sum_{\langle c_i, \mathbf{e}_i\rangle \in S} \log P_\Theta(\, c_i \,|\, \mathbf{e}_i\,) \tag{1}$$

rather than the ones that maximize $\sum_{\langle c_i, \mathbf{e}_i\rangle \in S} \log P_\Theta(\, c_i, \mathbf{e}_i\,)$ (Rip96).

Optimizing the LCL of the root node (given the other attributes) of a *naïve-bayes* structure can be formulated as a standard logistic regression problem (MN89; Jor95). General belief nets extend naïve-bayes-structures by permitting additional dependencies among the attributes. This paper provides a general discriminative learning tool `ELR` that can learn the parameters for an arbitrary structure, completing the analogy:

$$\text{NaïveBayes} \;:\; \text{General Belief Net} \;::\; \text{Logistic Regression} \;:\; \texttt{ELR}\,. \tag{2}$$

Moreover, while most algorithms for learning logistic regression functions require *complete* training data, the `ELR` algorithm can accept *incomplete* data. We also present empirical evidence, from a large number of datasets, to demonstrate that `ELR` works effectively.

Section 2 provides the foundations, overviewing belief nets then defining our task: discriminatively learning the parameters (for a fixed belief net structure, G) that maximize CL. Section 3 formally analyses this task, providing both sample and computational complexity, and noting how these results compare with corresponding results for generative learning. Seeing that our task is NP-hard in general, Section 4 presents a gradient-descent discriminative learning algorithm for general BNs, `ELR`. Section 5 reports empirical results that demonstrate that our `ELR` often produces a classifier that is often superior to ones produced by standard learning algorithms (which maximize likelihood), over a variety of situations: In particular, when the learner has complete data, we show that ELR can be superior to the standard "observed frequency estimate" (OFE) approach (CH92), and when given partial data, we show ELR can be more effective than the EM and APN systems.[1] As the ELR behavior can depend on how the given BN-structure G relates to the true structure T, we consider three regimes:

- when G is *simpler* than T — *i.e.*, when G has too few links;
- when G is *approximately equal* to T (*e.g.*, G is produced by a structure-learning algorithm);
- when G is *more complicated* than T — *i.e.*, when G has too many links.

Section 6 provides a brief survey of the relevant literature, and the appendix provides the proofs of our theoretic claims. The webpage (Gre04) provides more information about the experiments shown in Section 5, as well as other experiments.

2. Framework

We assume there is a stationary underlying distribution $P(\cdot)$ over N (discrete) random variables $\mathcal{V} = \{V_1, \ldots, V_n\}$; For example, perhaps V_1 is the "Cancer" random variable, whose value ranges over $\{$true, false$\}$; V_2 is "Gender" $\in \{$male, female$\}$, V_3 is "Age" $\in \{0, .., 100\}$, etc. We refer to this joint distribution as the "underlying distribution" or the "event distribution".

We can encode this as a "(Bayesian) belief net" (BN) — a directed acyclic graph $B = \langle \mathcal{V}, A, \Theta \rangle$, whose nodes $\mathcal{V}$ represent variables, and whose arcs A represent dependencies. Each node $D_i \in \mathcal{V}$ also includes a conditional-probability-table (CPtable) $\theta_i \in \Theta$ that specifies how D_i's values depend (stochastically) on the values of its immediate parents. In particular, given a node $D \in \mathcal{V}$ with immediate parents $\mathbf{F} \subset \mathcal{V}$, the parameter $\theta_{d|\mathbf{f}}$ represents the network's term for $P(D=d \mid \mathbf{F}=\mathbf{f})$ (Pea88).

The user interacts with the belief net by asking *queries*, each of the form "What is $P(C=c \mid \mathbf{E}=\mathbf{e})$?" — *e.g.*, What is $P($Cancer = true$\mid$Gender=female, Smoke=true$)$? — where $C \in \mathcal{V}$ is a single "query variable", $\mathbf{E} \subset \mathcal{V}$ is the subset of "evidence variables", and c (resp., $\mathbf{e}$) is a legal assignment to C (resp., $\mathbf{E}$). This paper focuses on the case where all queries involve the same variable; *e.g.*, all queries ask about Cancer. Moreover, we will follow standard practice by assuming the distribution of conditioning events matches the underlying distribution. This means there is a single distribution from which we can draw instances, which correspond to a set of labeled instances (aka "labeled queries").[2] Note this corresponds to the data sample used by standard learning algorithms.

[1] Section 5 provides an overview of these systems.
[2] See (GGS97) for an alternative position, and the challenges this requires solving.

Given any unlabeled instance $\{\mathbf{E}_i = \mathbf{e}_i\}$, the belief net[3] Θ will produce a distribution over the values of the query variable; perhaps $P_\Theta(\texttt{Cancer = true} \,|\, \mathbf{E} = \mathbf{e}) = 0.3$ and $P_\Theta(\texttt{Cancer = false} \,|\, \mathbf{E} = \mathbf{e}) = 0.7$. In general, the associated H_Θ classifier system will then return the value $H_\Theta(\mathbf{e}) = \text{argmax}_c\{P_\Theta(C = c \,|\, \mathbf{E} = \mathbf{e})\}$ with the largest posterior probability — here return $H_\Theta(\mathbf{E} = \mathbf{e}) =$ false as $P_\Theta(\texttt{Cancer = false} \,|\, \mathbf{E} = \mathbf{e}) > P_\Theta(\texttt{Cancer = true} \,|\, \mathbf{E} = \mathbf{e})$.

A good belief net classifier is one that produces the appropriate answers to these unlabeled queries. We will use "classification error" (aka "0/1" loss) to evaluate the resulting Θ-based classifier H_Θ

$$\text{err}(\Theta) \quad = \quad \sum_{\langle \mathbf{e}, c\rangle} P(\mathbf{e}, c) \; \times \; I(H_\Theta(\mathbf{e}) \neq c) \tag{3}$$

where $I(a \neq b) \; = \; 1$ if $a \neq b$, and $= 0$ otherwise.

Our goal is a belief net Θ^* that minimizes this score, with respect to the true distribution $P(\cdot)$. While we do not know this distribution *a priori*, we can use a sample drawn from this distribution, to help determine which belief net is optimal. This paper focuses on the task of learning the optimal CPtable Θ for a given BN-structure $G = \langle \mathcal{V}, A\rangle$.

Conditional Likelihood: In earlier work (Zho02, Sections 3.1.3 and 3.2.2), we compared learners that optimized $\text{err}(\Theta)$ versus ones that optimized the "log conditional likelihood" of a belief net Θ

$$\text{LCL}_P(\Theta) \quad = \quad \sum_{\langle \mathbf{e}, c\rangle} P(\mathbf{e}, c) \; \times \; \log(P_\Theta(c \,|\, \mathbf{e})) \tag{4}$$

(as approximated by Equation 1), and found no significant difference in classification performance; this is consistent with (MN89; FGG97; BKRK97). Our work therefore focuses on learners that attempt to maximize $\text{LCL}_P(\Theta)$.

While Equation 1's $\widehat{\text{LCL}}^{(S)}(\Theta)$ formula closely resembles the (empirical) "log likelihood" function

$$\widehat{\text{LL}}^{(S)}(\Theta) \quad = \quad \frac{1}{|S|} \sum_{\langle \mathbf{e}, c\rangle \in S} \log(P_\Theta(c, \mathbf{e})) \tag{5}$$

used by many BN-learning algorithms, there are some critical differences. As shown in (FGG97),

$$\widehat{\text{LL}}^{(S)}(\Theta) \; = \; \frac{1}{|S|} \left[\sum_{\langle c, \mathbf{e}\rangle \in S} \log(P_\Theta(c \,|\, \mathbf{e})) \; + \sum_{\langle c, \mathbf{e}\rangle \in S} \log(P_\Theta(\mathbf{e})) \right]$$

[3] As we assume the structure $G = \langle \mathcal{V}, A\rangle$ of the belief net $B = \langle \mathcal{V}, A, \Theta\rangle$ is fixed, we will identify a belief net with its parameters Θ.

where the first term resembles our $\widehat{\mathrm{LCL}}(\cdot)$ score, which measures how well our network will answer the relevant queries, while the second term is irrelevant to our task. This means a BN Θ_α that does poorly wrt the first "$\widehat{\mathrm{LCL}}(\cdot)$-like" term may be preferred to a Θ_β that does better — *i.e.*, it is possible that $\widehat{\mathrm{LL}}(\Theta_\alpha) > \widehat{\mathrm{LL}}(\Theta_\beta)$, while $\widehat{\mathrm{LCL}}(\Theta_\alpha) < \widehat{\mathrm{LCL}}(\Theta_\beta)$. (Section 5.4 provides other arguments explaining why our $\widehat{\mathrm{LCL}}(\cdot)$-based approach may work better than the $\widehat{\mathrm{LL}}(\cdot)$-based approaches; and Section 6 surveys other relevant literature.)

Finally, all of our algorithms are producing parameters for a given belief net structure, G. We therefore define $\mathcal{BN}(G) = \{\langle G, \Theta\rangle\}$ to be the set of all belief nets that use the structure G; our goal is a setting for the CPtable parameters $\Theta \in [0,1]^k$, appropriate for this G structure, that produces the optimal classification performance. We view $\mathcal{BN}(G)$ as the set of these parameters.

3. Theoretical Analysis

How many "labeled instances" are enough — *i.e.*, given any values $\varepsilon, \delta > 0$, how many labeled instances are needed to insure that, with probability at least $1-\delta$, an algorithm can produce a classifier that is within ε of optimal? While we believe there are comprehensive general bounds, our specific results require the relatively benign technical restriction that all CPtable entries must be bounded away from 0. That is, for any $\gamma > 0$, let

$$\mathcal{BN}_{\Theta\succeq\gamma}(G) \quad = \quad \{\Theta \in \mathcal{BN}(G) \mid \forall \theta_{d|\mathbf{f}} \in \Theta,\ \theta_{d|\mathbf{f}} \geq \gamma\} \tag{6}$$

be the subset of BNs whose CPtable values are all at least γ.[4] We now restrict our attention to these belief nets, and in particular, let

$$\Theta^*_{G,\Theta\succeq\gamma} \quad = \quad \operatorname{argmax}\left\{\mathrm{LCL}_P(\Theta) \mid \Theta \in \mathcal{BN}_{\Theta\succeq\gamma}(G)\right\} \tag{7}$$

be a BN with optimal score among $\mathcal{BN}_{\Theta\succeq\gamma}(G)$ with respect to the true distribution $P(\cdot)$.

THEOREM 1. *Let G be any belief net structure with K CPtable entries $\Theta = \{\theta_{d_i|\mathbf{f}_i}\}_{i=1..K}$, and let $\hat{\Theta} \in \mathcal{BN}_{\Theta\succeq\gamma}(G)$ be the BN in $\mathcal{BN}_{\Theta\succeq\gamma}(G)$ that has maximum* empirical *log conditional likelihood score (Equation 1) with respect to a sample of*

$$M_{\gamma,N,K}(\varepsilon,\delta) \;=\; 18\left(\frac{N\ln\gamma}{\varepsilon}\right)^2\left[\ln\frac{2}{\delta} \;+\; K\ln\frac{6K}{\gamma\varepsilon}\right] \;=\; O\left(\frac{N^2K}{\varepsilon^2}\ln(\frac{K}{\varepsilon\delta})\ \ln^3(\frac{1}{\gamma})\right) \tag{8}$$

[4] This $\theta_{d|\mathbf{f}} \geq \gamma$ constraint is trivially satisfied by any parameter learner that uses Laplacian correction, or that produces the posterior distribution from uniform Dirichlet priors: These system can use $\gamma = 1/(m+2)$ where m is the number of training instances (Hec98).

labeled queries drawn from $P(\cdot)$. *Then, with probability at least* $1-\delta$, $\hat{\Theta}$ *will be no more than* ε *worse than* $\Theta^*_{G,\Theta\succ\gamma}$ *— i.e.,* $P(\,LCL_P(\,\hat{\Theta}\,) \leq LCL_P(\,\Theta^*_{G,\Theta\succeq\gamma}\,) - \varepsilon\,) \leq \delta$. ∎

A virtually identical proof shows that this same result holds when dealing with another approximation to the 0/1 error (err$(\,\Theta\,)$),

$$MSE(\Theta) = \sum_{\langle \mathbf{e},c\rangle} P(\,\mathbf{e},\,c\,) \;\times\; [P_\Theta(\,c\,|\,\mathbf{e}\,) - P(\,c\,|\,\mathbf{e}\,)]^2 \tag{9}$$

rather than LCL$(\,\cdot\,)$.

This *PAC*-learning (Val84) result can be used to bound the learning rate — *i.e.*, for a fixed structure G and confidence term δ, it specifies how many instances M are required to guarantee an additive error of at most ε — note the $O(\frac{1}{\varepsilon^2}[\log\frac{1}{\varepsilon}])$ dependency.[5]

For comparison, Dasgupta (Das97, Section 5) proves that

$$\frac{288N^2K}{\varepsilon^2}\ln^2\left(1+\frac{3N}{\varepsilon}\right)\ln\frac{18NK\ln(1+3N/\varepsilon)}{\varepsilon\delta} \;=\; O\left(\frac{N^2K}{\varepsilon^2}\,\ln(\frac{K}{\varepsilon\delta})\;\ln^3(N)\,\ln^2(\frac{1}{\varepsilon})\right) \tag{10}$$

complete tuples[6] are sufficient to learn the parameters to a fixed structure that are with ε of the optimal *likelihood* (Equation 5). While comparing upper bounds is only suggestive, it is interesting to note that, ignoring the $\ln^\ell(\cdot)$ terms for $\ell > 1$, these bounds are asymptotically identical.

One asymmetry is that only our Equation 8 bound includes the γ term, which corresponds to the smallest CPtable entry allowed. While (Das97) (following (ATW91)) can avoid this term by "tilting" the empirical distribution, this trick does not apply in our discriminative task: Our task inherently involves computing *conditional* likelihood, which requires *dividing* by some CPtable values, which is problematic when these values are near 0. This observation also means our proof is *not* an immediate application of the standard *PAC*-learning approaches. Of course, our sample complexity remains polynomial in the size (N, K) of the belief net even if this γ is exponentially small, $\gamma = O(1/2^N)$.

Note finally that the parameters that optimize (or nearly optimize) likelihood will not necessarily optimize our objective of *conditional* likelihood; this means Equation 10 describes the convergence to parameters that are typically inferior to the ones associated with Equation 1, especially when the structure is wrong; see (NJ01).

---

[5] These are still very pessimistic bounds. For example, using the modest $\varepsilon = \delta = \lambda = 0.05$ setting, this would require ≈4.77 billion training examples to deal with the VOTE dataset!

[6] We say a tuple is "complete" if it specifies a value for every attribute; hence "$E_1 = e_1, \ldots, E_n = e_n$" is complete (where $\{E_1, \ldots, E_n\}$ is the full set of evidence variables) but "$E_2 = e_2, E_7 = e_7$" is not.

The second question is computational: How hard is it to find these best parameters values, given this sufficiently large sample. Here, the news is mixed.

On the positive side, Wettig *et al.* (WGR$^+$03) show this task corresponds to a convex optimization problem when the data is complete and the structure G satisfies certain specified properties; this implies a polynomial algorithm can find the values of the CPtables of G whose (empirical) conditional likelihood (Equation 1) is within ε of optimal (NN94; BV04). They show that these properties hold for NaïveBayes and TAN structures (defined in Section 5 below).

Unfortunately...

THEOREM 2. *It is NP-hard to find the values for the CPtables of a fixed BN-structure that produce the largest (empirical) conditional likelihood (Equation 1) for a given* incomplete *sample.* ∎

We do not know the complexity of this task for *arbitrary* structures, given complete data.

4. ELR Learning Algorithm

Given the intractability of computing the optimal CPtable entries in general, we defined a simple gradient-ascent algorithm, ELR, that attempts to improve the empirical score $\widehat{\mathrm{LCL}}(\Theta)$ by changing the values of each CPtable entry $\theta_{d|\mathbf{f}}$. (Of course, this will only find, at best, a *local* optimum.) To incorporate the constraints $\theta_{d|\mathbf{f}} \geq 0$ and $\sum_d \theta_{d|\mathbf{f}} = 1$, we used the different set of parameters, "$\beta_{d|\mathbf{f}}$", where each

$$\theta_{d|\mathbf{f}} \quad = \quad \frac{e^{\beta_{d|\mathbf{f}}}}{\sum_{d'} e^{\beta_{d'|\mathbf{f}}}} \,. \tag{11}$$

As the β_is sweep over the reals, the corresponding $\theta_{d_i|\mathbf{f}}$'s will satisfy the appropriate constraints. (In the naïve-bayes case, this corresponds to what many logistic regression algorithms would do, albeit with different parameters (Jor95): Find α, χ that optimize $P_{\alpha,\chi}(C{=}c\,|\,\mathbf{E}{=}\mathbf{e}) \;=\; e^{\alpha_c + \chi_c \cdot \mathbf{e}} / \sum_j e^{\alpha_j + \chi_j \cdot \mathbf{e}}$.[7] Recall that our goal is a more general algorithm — one that can deal with *arbitrary* structures; see Equation 2.)

Like all such algorithms, ELR is basically

$$\begin{array}{l} \text{Initialize } \beta^{(0)} \\ \text{For } k = 1..m \\ \qquad \beta^{(k+1)} \quad := \quad \beta^{(k)} \,+\, \alpha^{(k)} \times \mathbf{d}^{(k)} \end{array} \tag{12}$$

[7] While the obvious tabular representation of the CPtables involves more parameters than appear in this logistic regression model, these extra BN-parameters are redundant.

where $\beta^{(k)}$ represents the set of parameters at iteration k. In a simple gradient ascent approach, we would set $\mathbf{d}^{(k)}$ to be the total derivative with respect to the given set of labeled queries, $\nabla\widehat{LCL} = \langle \frac{\partial\, \widehat{\mathrm{LCL}}^{(S)}(\Theta)}{\partial\, \beta_{d|\mathbf{f}}} \rangle_{d,\mathbf{f}}$, which is the sum of the individual derivatives from each labeled training case $\langle \mathbf{e}; c \rangle$:

$$\frac{\partial\, \widehat{\mathrm{LCL}}^{(S)}(\Theta)}{\partial\, \beta_{d|\mathbf{f}}} = \sum_{\langle \mathbf{e},c\rangle \in S} \frac{\partial\, \widehat{\mathrm{LCL}}^{(\langle \mathbf{e},c\rangle)}(\Theta)}{\partial\, \beta_{d|\mathbf{f}}} .$$

This requires...

PROPOSITION 3. *For the labeled training case* $\langle \mathbf{e}, c\rangle$ *and each "softmax" parameter* $\beta_{d|\mathbf{f}}$,

$$\frac{\partial\, \widehat{LCL}^{(\langle \mathbf{e},c\rangle)}(\Theta)}{\partial\, \beta_{d|\mathbf{f}}} = [P_\Theta(d,\mathbf{f}|\,\mathbf{e},c) - P_\Theta(d,\mathbf{f}|\,\mathbf{e})] - \theta_{d|\mathbf{f}}\,[P_\Theta(\mathbf{f}|\,c,\mathbf{e}) - P_\Theta(\mathbf{f}|\,\mathbf{e})] .$$

Recall $P_\Theta(x\,|\,y)$ refers to conditional probability of x given a particular value of y, for the parameter setting corresponding to $\Theta \sim \langle \beta_{d|\mathbf{f}} \rangle$. Notice this expression is well-defined for any set of evidence variables $\mathbf{E}$ — which can be all "non-C" variables (corresponding to a complete data tuple), or any subset, including the empty $\mathbf{E} = \{\}$.

To work effectively, `ELR` incorporates four instantiations/modifications to the basic gradient ascent idea, dealing with

1. the initial values $\beta^{(0)}$ ("plug-in parameters"),
2. the direction of the modification $\mathbf{d}^{(k)}$ (conjugate gradient),
3. the magnitude of the change $\alpha^{(k)}$ (line search) and
4. the stopping criteria m ("cross tuning").

Minka (Min01) has shown that the middle two ideas, conjugate gradient and line-search (PFTV02), are effective for the standard logistic regression task.

Begin with Plug-In Parameters: We must first initialize the parameters, $\beta^{(0)}$. One common approach is to set $\beta^{(0)}$ to small, randomly selected, values; another is to begin with the values specified by the generative approach — *i.e.*, using frequency estimates (`OFE`; Equation 13) in the complete data case, and a simple variant otherwise.[8] Our empirical evidence shows that this

[8] To compute the value for the $\beta_{d|\mathbf{f}}$ parameter, use frequency estimates but only over the training instances that specify the values of all $\{D\} \cup \mathbf{F}$ variables.

second approach works better, especially for small samples. These easy-to-compute generative starting values are often used to initialize parameters for discriminative tasks, and called “plug-in parameters” (Rip96).

Conjugate Gradient: Standard gradient ascent algorithms may require a great many iterations, especially for functions that have long, narrow valley structures. The *conjugate gradient method* addresses this problem by ascending along *conjugate directions*, rather than simply the local gradient. As this means that the directions that have already been optimized, stay optimized, this method often requires far fewer steps uphill to reach the local optimum (HDB96).

In particular, ELR uses the *Polak-Rebiere* formula to update its search direction. The initial search direction is given by:

$$\mathbf{d}^{(0)} \quad = \quad \nabla\widehat{LCL}_{(0)}$$

On subsequent iterations,

$$\mathbf{d}^{(k)} \quad = \quad \nabla\widehat{LCL}_{(k)} \; - \; \frac{(\nabla\widehat{LCL}_{(k)} - \nabla\widehat{LCL}_{(k-1)}) \;\cdot\; \nabla\widehat{LCL}_{(k)}}{\nabla\widehat{LCL}_{(k-1)} \;\cdot\; \nabla\widehat{LCL}_{(k-1)}}\, \mathbf{d}^{(k-1)}$$

where the “$\cdot$” is the vector dot-product. Hence, the current update direction is formed by “subtracting off” the previous direction from the current derivative.

Line Search: ELR will ascend in the $\mathbf{d}^{(k)}$ direction; the next challenge is deciding how far to move — *i.e.*, in computing the $\alpha^{(k)} \in \Re^{+}$ from Equation 12.

The good news is, as $\beta = \beta^{(k)}$ and $\mathbf{d} = \mathbf{d}^{(k)}$ are fixed, this is a one-dimensional search: first α^{*} that maximizes $f(\alpha) = \widehat{\mathrm{LCL}}^{(S)}(\beta \;+\; \alpha_i \times \mathbf{d})$. ELR therefore uses a standard technique, *Brent*'s iterative line search procedure (a hybrid combination of the linear Golden Section search (HDB96) and a quadratic interpolation), which has proven to be very effective at finding the optimal value for $\alpha^{(k)}$ (Bis98). In essense, this method first finds three α_i values, and computes the associated function values $\langle\, \langle \alpha_i,\; f(\alpha_i)\rangle \,\rangle_{i=1,2,3}$. It then assumes the $f(\cdot)$ function is (locally) quadratic, and so fits this trio of points to a second degree polynomial. It then finds the α^{*} value that minimizes this quadratic, replaces one of the three original α_i values with this $\langle \alpha^{*}, f(\alpha^{*})\rangle$ pair, and iterates using the new trio of points. See (Bis98) for details.

Cross tuning (stopping time): The final issue is deciding when to stop; *i.e.*, determining the value of *m* for Equation 12. A naïve algorithm would just compute the training-set error (Equation 3) at each iteration, and stop when that error measure appears at a local optimum — *i.e.*, when the error appears to be going up. The graph in Figure 1 shows both training-set and test-set error on a particular dataset (here “CLEVE”; see next section), at each iteration. If we used this simple approach, we would stop on the third iteration; the

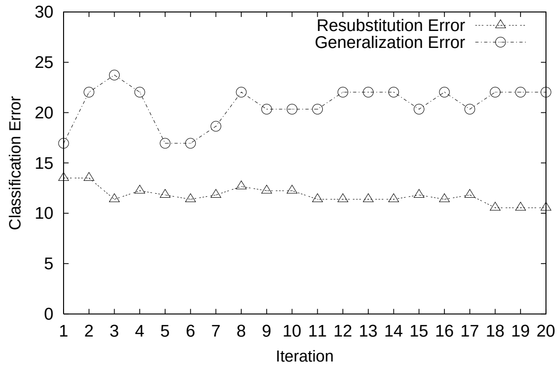

Figure 1. Comparing Training-Set Error with Test-Set Error, as function of Iteration

generalization error shows that these parameters are very bad — in fact, they seem almost the worst values!

To avoid this, we use a variant of cross validation, which we call "cross tuning", to estimate the optimal number of iterations. Here, we divide the training data into $\ell = 5$ partitions, then for each fold, run the gradient ascent for remaining $4/5$ of the data, but evaluating the quality of the result (on each iteration) on the fold. (This produces a graph like the "Generalization Error" line in Figure 1.) We then determine, for each fold, when we should have stopped — here, it would be on $k = 5$, as that is the global optimum for this fold. We then set m to be the median values over these folds. When using all of the data to produce the final $\beta_{d|\mathbf{f}}$ values, we iterate exactly m times. The website (Gre04, #CrossTuning) presents empirical evidence that cross-tuning is important, especially for complex models.

5. Empirical Studies

The `ELR` algorithm takes, as arguments, a BN-structure $G = \langle \mathcal{V}, A \rangle$ and a dataset of labeled queries (aka instances) $S = \{\langle \mathbf{e}_i, c_i \rangle\}_i$, and returns a value for each CPtable parameter $\theta_{d|\mathbf{f}}$. To explore its effectiveness, we compared the $\text{err}(\cdot)$ performance of the resulting Θ_{ELR} parameters with the results of other algorithms that similarly learn CPtable values for a given structure.

We say a data sample is "complete" if each instance is complete (see Footnote 6); otherwise it is incomplete. When the data is complete, we compare `ELR` to the standard "observed frequency estimate" (`OFE`) approach, which is known to produce the parameters that maximize likelihood (Equation 5) for a given structure (CH92). For example, if 75 of the 100 $C = 1$ instances have

$X_3 = 0$, then OFE sets

$$\theta_{X_3=0|C=1} \quad = \quad \frac{75}{100} \tag{13}$$

(Some versions use a Laplacian correction to avoid the problems caused by 0 probability events.) When the data is *incomplete*, we compare ELR to the standard Expectation Maximization algorithm EM (DLR77; Lau95; Hec98) and to APN (BKRK97), which ascends to parameter values whose likelihood is locally optimal.

Here we present only the results of the ELR=ELR_β algorithm, that used the β terms (Equation 11), as we found its performance strictly dominated the ELR_θ version that used θ directly. Similarly, while the original APN_θ (BKRK97) climbed in the space of parameters $\Theta = \{\theta_i\}$, we instead used a modified APN_β system that uses the $\beta = \{\beta_i\}$ values, as we found it worked better as well.

Traditional wisdom holds that discriminative learning (ELR) is most relevant (*i.e.*, better than generative learning: OFE, APN, EM) when this underlying model $G = \langle \mathcal{V}, A \rangle$ is "wrong", that is, not an I-map of the true distribution T — which here means the graph structure does not include some essential arcs (Pea88). The situation, which we denote "$G < T$", is fairly common as many learners consider only structures as simple as naïve-bayes, or the class of TAN structures (defined below), which are typically much simpler than T.[9] Section 5.1 deals with this situation, considering both the complete and incomplete data cases.

Section 5.2 then considers another standard situation: where we employ a structure-learning algorithm to produce a structure that is similar to the truth; *i.e.*, where $G \approx T$. Here, we use the PowerConstructor system (CGK02; CG99) for the first step (to learn the structure), then compare the relative effectiveness of algorithms for finding parameters for this structure.

We next consider the uncommon situation where the model G is more *complicated* than the truth T; *i.e.*, $G > T$. Section 5.3 uses artificial data to compare ELR vs OFE, APN and EM, in this context.

Finally, Section 5.4 summarizes all of these empirical results.

Notation: The notation "NB+ELR" will refer to the NB structure, whose parameters are learned using ELR; in general, we will use *x*+*y* to refer to the system produced when the *y* algorithm is used to produce the parameter for the *x* structure. Below we will compare various pairs of learners, in each case over a common dataset; we will therefore use a one-sided paired t-test (Mit97). When this result is significant at the $\rho < 0.05$ level, we will write $\alpha \Leftarrow_{(p<\rho)} \beta$ — *e.g.*, we will soon see NB+ELR$\Leftarrow_{(p<0.005)}$NB+OFE. For values larger than 0.05, we will use the notation $\alpha \leftarrow_{(p<\rho)} \beta$. (That is, we regard $p < 0.05$ as the

[9] Note the $G < T$ notation does *not* mean the arcs of G are a subset of T's, as G may also include arcs that are not in T.

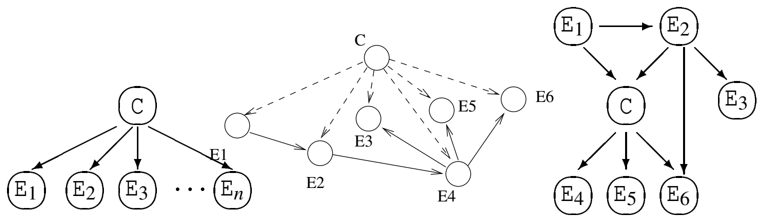

Figure 2. (a) NaïveBayes structure (b) TAN structure (FGG97, Fig 3) (c) Arbitrary GBN structure. (Note: all figures are numbered (a), (b), ... from left to right.)

cut-off for statistical significance.) Note the arrow points to the learner that appears to be better.[10]

While our main emphasis is comparing x+ELR to x+OFE (and to x+APN and x+EM) for various structures x, where relevant we will also compare across structure classes; *e.g.*, comparing x+ELR to y+ELR for different structures x and y.

5.1. Model is Simpler than the Truth ($G < T$)

Section 5.1.1 (resp., Section 5.1.2) compares algorithms for learning the parameters for a naïve-bayes model (resp., a TAN model) given *complete* data; Section 5.1.3 then considers learning these models given *incomplete* data. Section 5.1.4 uses a simple controlled study, on artificial data, to further investigate a specific claim.

5.1.1. *NaïveBayes — Complete, Real World Data*

Our first experiments deal 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 2(a). It is well-known that ELR, with this structure, corresponds to standard logistic regression (NJ01).

We compared the relative effectiveness of ELR with various other classifiers, over the same 25 datasets that (FGG97) used for their comparisons: 23 from UCIrvine repository (BM00), plus "MOFN-3-7-10" and "CORRAL", which were developed by (KJ97) to study feature selection; see Table I, which also specifies how we computed our accuracy values — based on 5-fold cross validation for small data, and holdout method for large data (Koh95). To deal with continuous variables, we implemented supervised entropy discretization (FI93). Table II summarizes the results.

We use the CHESS dataset (36 binary or ternary attributes) to illustrate the basic behaviour of the algorithms. Figure 3(a) shows the performance,

[10] This analysis makes the standard assumptions that the datapoints are identical and independent, each normally distributed; see (NB03).

Table I. Description of data sets used in the experiments (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

on this dataset, of our NB+ELR (a.k.a. "NaïveBayes structure + ELR instantiation") system, versus the "standard" NB+OFE, which uses OFE to instantiate the parameters. We see that ELR is consistently more accurate than OFE, for any size training sample. We also see how quickly ELR converges to the best performance.

Figure 4(a) provides a more comprehensive comparison, across all 25 datasets. (Each point below the $x = y$ line is a dataset where NB+ELR was better than other approach — here NB+OFE. The lines also express the 1 standard-

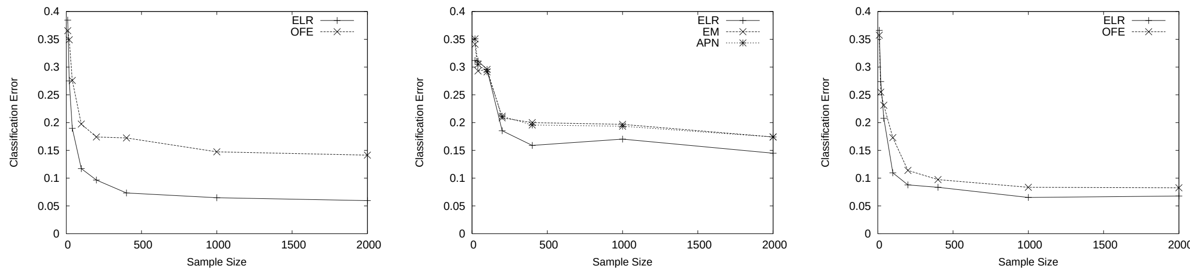

Figure 3. CHESS domain: (a) ELR vs OFE, complete data, structure is "incorrect" (naïve-bayes); (b) ELR vs EM, APN on incomplete data, structure is "incorrect" (naïve-bayes) (c) ELR vs OFE, complete data, structure is "≈correct" (POWERCONSTRUCTOR)

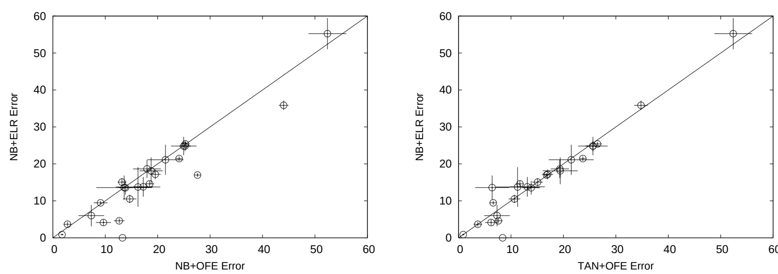

Figure 4. Comparing NB+ELR with (a) NB+OFE (b) TAN+OFE

deviation error bars in each dimension.[11]) As suggested by this plot, NB+ELR is significantly better than NB+OFE at the $p < 0.005$ level.

5.1.2. *TAN — Complete, Real World Data*

We next considered TAN ("tree augmented naïve-bayes") structures (FGG97), which include a link from the classification node down to each attribute and, if we ignore those class-to-attribute links, the remaining links, connecting attributes to each other, form a tree; see Figure 2(b). (Hence this representation allows each attribute to have at most one "attribute parent", and so this class of structures strictly generalize NaïveBayes.) Friedman *et al.* (FGG97) provide an efficient algorithm for learning such TAN structures given complete data, based on (CL68): first compute the mutual information between each pair of attributes, conditioned on the class variable, then find the minimum-weighted spanning tree within this complete graph of the attributes. (Each mutual information quantity is based on the empirical sample.) They prove that the resulting structure maximizes the likelihood of the data, over all possible TAN structures — *n.b.*, this is optimizing a generative measure.

[11] When using 5-fold cross-validation, we computed the standard deviation using the 5 computed accuracy values. When dealing with a single split of the data, we used the standard binomial formula, $\sqrt{p \times (1-p)/n}$, where p is the accuracy and n is the size of the test set.

Table II. Empirical accuracy of classifiers learned from *complete* data

	Data set	NB+OFE	NB+ELR	TAN+OFE	TAN+ELR	GBN+OFE	GBN+ELR
1	AUSTRALIAN	86.81±0.84	84.93±1.06	84.93±1.03	84.93±1.03	86.38±0.98	86.81±1.11
2	BREAST	97.21±0.75	96.32±0.66	96.32±0.81	96.32±0.70	96.03±0.50	95.74±0.43
3	CHESS	87.34±1.02	95.40±0.64	92.40±0.81	97.19±0.51	90.06±0.92	90.06±0.92
4	CLEVE	82.03±2.66	81.36±2.46	80.68±1.75	81.36±1.78	84.07±1.48	82.03±1.83
5	CORRAL	86.40±5.31	86.40±3.25	93.60±3.25	100.00±0.00	100.00±0.00	100.00±0.00
6	CRX	86.15±1.29	86.46±1.85	86.15±1.70	86.15±1.70	86.00±1.94	85.69±1.30
7	DIABETES	74.77±1.05	75.16±1.39	74.38±1.35	73.33±1.97	75.42±0.61	76.34±1.30
8	FLARE	80.47±1.03	82.82±1.35	83.00±1.06	83.10±1.29	82.63±1.28	82.63±1.28
9	GERMAN	74.70±0.80	74.60±0.58	73.50±0.84	73.50±0.84	73.70±0.68	73.70±0.68
10	GLASS	47.62±3.61	44.76±4.22	47.62±3.61	44.76±4.22	47.62±3.61	44.76±4.22
11	GLASS2	81.25±2.21	81.88±3.62	80.63±3.34	80.00±3.90	80.63±3.75	78.75±3.34
12	HEART	78.89±4.08	78.52±3.44	78.52±4.29	78.15±3.86	79.63±3.75	78.89±4.17
13	HEPATITIS	83.75±4.24	86.25±5.38	88.75±4.15	85.00±5.08	90.00±4.24	90.00±4.24
14	IRIS	92.67±2.45	94.00±2.87	92.67±2.45	92.00±3.09	92.00±3.09	92.00±3.09
15	LETTER	72.40±0.63	83.02±0.53	83.22±0.53	88.90±0.44	79.78±0.57	81.21±0.55
16	LYMPHOGRAPHY	82.76±1.89	86.21±2.67	86.90±3.34	84.83±5.18	79.31±2.18	78.62±2.29
17	MOFN-3-7-10	86.72±1.06	100.00±0.00	91.60±0.87	100.00±0.00	86.72±1.06	100.00±0.00
18	PIMA	75.03±2.45	75.16±2.48	74.38±2.81	74.38±2.58	75.03±2.25	74.25±2.53
19	SATIMAGE	81.55±0.87	85.40±0.79	88.30±0.72	88.30±0.72	79.25±0.91	79.25±0.91
20	SEGMENT	85.32±1.28	89.48±1.11	89.35±1.11	89.22±1.12	77.53±1.50	77.40±1.51
21	SHUTTLE-SMALL	98.24±0.30	99.12±0.21	99.12±0.21	99.22±0.20	97.31±0.37	97.88±0.33
22	SOYBEAN-LARGE	90.89±1.31	90.54±0.54	93.39±0.67	92.86±1.26	82.50±1.40	85.54±0.99
23	VEHICLE	55.98±0.93	64.14±1.28	65.21±1.32	66.39±1.22	48.52±2.13	51.95±1.32
24	VOTE	90.34±1.44	95.86±0.78	93.79±1.18	95.40±0.63	96.32±0.84	95.86±0.78
25	WAVEFORM-21	75.91±0.62	78.55±0.60	76.30±0.62	76.30±0.62	65.79±0.69	65.79±0.69

Figure 4(b) compares NB+ELR to TAN+OFE. We see that ELR, even when handicapped with the simple NB structure, performs about as well as OFE on TAN structures. Of course, the limitations of the NB structure may explain the poor performance of NB+ELR on some data. For example, in the CORRAL dataset, as the class is a function of four interrelated attributes, one must connect these attributes to predict the class. As NaïveBayes permits no such connection, NaïveBayes-based classifiers performed 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+ELR is still comparable to TAN+OFE, in general.

Would we do yet better by using ELR to instantiate TAN structures? While Figure 5(a) suggests that TAN+ELR is slightly better than NB+ELR, this is

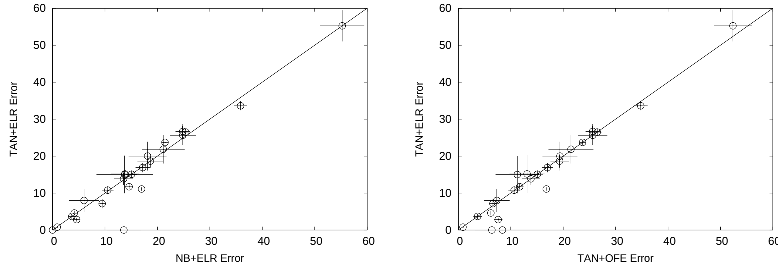

Figure 5. Complete data: Comparing `TAN+ELR` vs (a) `NB+ELR` (b) `TAN+OFE`

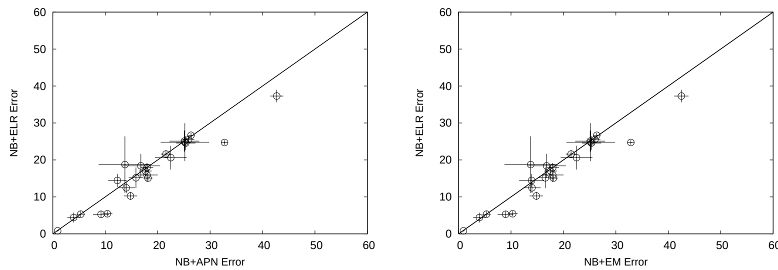

Figure 6. Incomplete data: `NB+ELR` vs (a) `NB+APN`; (b) `NB+EM`

not significant: only at the $p < 0.2$ level. However, Figure 5(b) shows that `TAN+ELR` does consistently better than `TAN+OFE` — at a $p < 0.025$ level. We found that `TAN+ELR` did perfectly on the the CORRAL dataset, which `NB+ELR` found problematic.

5.1.3. `NB`, `TAN` — *Incomplete, Real World Data*

All of the above studies used *complete* data. We next explored how well `ELR` 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. (Hence, this data is "missing completely at random", MCAR (LR87).) We then compared `ELR` to the standard "missing-data" learning algorithms, `APN` and `EM`. In each case — for `ELR`, `APN` and `EM` — we initialize the parameters using the obvious variant of `OFE` that considers, for $\beta_{d|\mathbf{f}}$, only the records that include values for the relevant node and all of its parents $\{D\} \cup \mathbf{F}$.

Here, we first learned the parameters for the NaïveBayes structure; Figure 3(b) shows the learning curve for the CHESS domain, comparing `ELR` to `APN` and `EM`. We see that `ELR` does better for essentially every sample size. We also compared these algorithms over the rest of the 25 datasets; see Figures 6(a) and 6(b) for `ELR` vs `APN` and `ELR` vs `EM`, respectively. As shown, `ELR` does consistently better — in each case, at the $p < 0.025$ level.

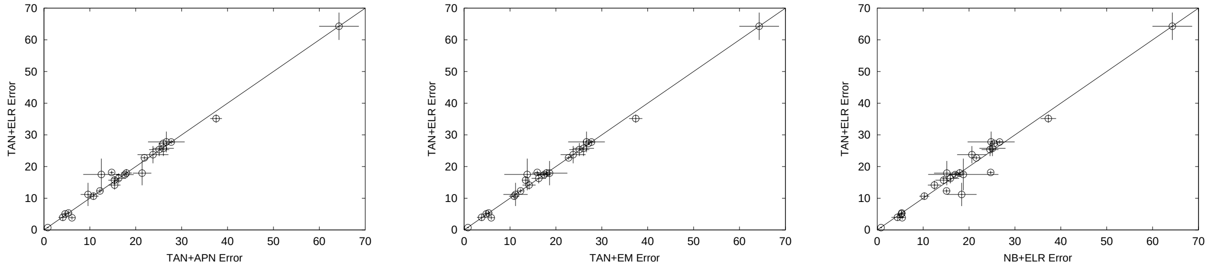

Figure 7. Incomplete data: Comparing TAN+ELR with (a) TAN+APN; (b) TAN+EM; (c) NB+ELR

Table III. Empirical accuracy of classifiers learned from *incomplete* data

Data set	NB+ELR	NB+APN	NB+EM	TAN+ELR	TAN+APN	TAN+EM	GBN+ELR	GBN+APN	GBN+EM
AUSTRALIAN	$78.41_{\pm1.01}$	$78.41_{\pm0.96}$	$78.55_{\pm1.01}$	$77.25_{\pm0.59}$	$78.12_{\pm0.74}$	$77.25_{\pm0.59}$	$74.06_{\pm1.06}$	$74.06_{\pm1.06}$	$74.78_{\pm0.74}$
BREAST	$95.59_{\pm1.32}$	$96.03_{\pm1.20}$	$96.03_{\pm1.20}$	$96.03_{\pm1.13}$	$95.88_{\pm0.95}$	$96.18_{\pm1.02}$	$94.12_{\pm1.63}$	$94.85_{\pm1.36}$	$94.85_{\pm1.36}$
CHESS	$94.56_{\pm0.69}$	$89.59_{\pm0.94}$	$89.68_{\pm0.93}$	$96.15_{\pm0.59}$	$93.90_{\pm0.73}$	$94.09_{\pm0.72}$	$90.34_{\pm0.90}$	$90.06_{\pm0.92}$	$90.06_{\pm0.92}$
CLEVE	$84.07_{\pm1.90}$	$82.03_{\pm2.05}$	$82.03_{\pm2.05}$	$83.73_{\pm1.57}$	$83.73_{\pm1.57}$	$83.73_{\pm1.57}$	$83.05_{\pm1.93}$	$81.36_{\pm2.34}$	$83.39_{\pm1.89}$
CORRAL	$81.60_{\pm3.25}$	$83.20_{\pm3.67}$	$83.20_{\pm3.67}$	$88.80_{\pm3.67}$	$90.40_{\pm1.60}$	$88.80_{\pm2.65}$	$92.00_{\pm1.79}$	$88.80_{\pm2.65}$	$92.00_{\pm1.79}$
CRX	$87.54_{\pm1.43}$	$86.00_{\pm1.67}$	$86.00_{\pm1.67}$	$85.85_{\pm1.43}$	$84.62_{\pm1.29}$	$85.85_{\pm1.43}$	$86.15_{\pm1.67}$	$87.23_{\pm1.10}$	$86.92_{\pm0.97}$
DIABETES	$75.42_{\pm1.84}$	$74.64_{\pm1.83}$	$74.64_{\pm1.83}$	$74.64_{\pm2.06}$	$74.90_{\pm2.19}$	$74.90_{\pm2.19}$	$73.46_{\pm1.99}$	$73.20_{\pm1.99}$	$72.81_{\pm1.79}$
FLARE	$83.00_{\pm1.42}$	$82.35_{\pm1.21}$	$82.44_{\pm1.24}$	$82.54_{\pm0.86}$	$82.35_{\pm1.90}$	$82.54_{\pm1.52}$	$82.63_{\pm1.28}$	$82.63_{\pm1.28}$	$82.63_{\pm1.28}$
GERMAN	$74.50_{\pm0.89}$	$74.10_{\pm1.09}$	$74.00_{\pm1.05}$	$72.70_{\pm0.54}$	$74.00_{\pm0.97}$	$72.90_{\pm0.40}$	$73.70_{\pm0.68}$	$73.40_{\pm0.86}$	$73.70_{\pm0.68}$
GLASS	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$	$35.71_{\pm4.33}$
GLASS2	$79.38_{\pm3.22}$	$77.50_{\pm3.03}$	$77.50_{\pm3.03}$	$76.25_{\pm2.72}$	$76.25_{\pm3.37}$	$76.25_{\pm2.72}$	$78.13_{\pm3.28}$	$77.50_{\pm3.75}$	$78.13_{\pm3.28}$
HEART	$75.19_{\pm5.13}$	$74.81_{\pm4.63}$	$74.81_{\pm4.63}$	$72.22_{\pm3.26}$	$73.33_{\pm4.00}$	$73.33_{\pm4.00}$	$73.70_{\pm3.95}$	$73.33_{\pm4.37}$	$73.33_{\pm4.37}$
HEPATITIS	$81.25_{\pm7.65}$	$86.25_{\pm5.00}$	$86.25_{\pm5.00}$	$82.50_{\pm5.00}$	$87.50_{\pm3.95}$	$86.25_{\pm5.00}$	$86.25_{\pm3.64}$	$86.25_{\pm3.64}$	$86.25_{\pm3.64}$
IRIS	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$	$94.67_{\pm0.82}$
LETTER	$75.28_{\pm0.61}$	$67.24_{\pm0.66}$	$67.14_{\pm0.66}$	$81.86_{\pm0.54}$	$85.25_{\pm0.50}$	$84.07_{\pm0.52}$	$72.80_{\pm0.63}$	$69.81_{\pm0.65}$	$68.60_{\pm0.66}$
LYMPHOGRAPHY	$84.83_{\pm2.80}$	$84.14_{\pm1.38}$	$83.45_{\pm1.29}$	$82.07_{\pm3.84}$	$78.62_{\pm2.01}$	$81.38_{\pm3.87}$	$78.62_{\pm2.29}$	$78.62_{\pm2.29}$	$79.31_{\pm2.18}$
MOFN-3-7-10	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$	$82.03_{\pm1.20}$
PIMA	$74.90_{\pm2.85}$	$74.90_{\pm2.85}$	$74.90_{\pm2.85}$	$74.25_{\pm2.45}$	$73.99_{\pm2.28}$	$73.99_{\pm2.28}$	$73.99_{\pm2.06}$	$74.64_{\pm2.25}$	$74.77_{\pm2.31}$
SATIMAGE	$84.90_{\pm0.80}$	$81.85_{\pm0.86}$	$81.90_{\pm0.86}$	$87.70_{\pm0.73}$	$87.80_{\pm0.73}$	$87.70_{\pm0.73}$	$73.95_{\pm0.98}$	$76.35_{\pm0.95}$	$76.30_{\pm0.95}$
SEGMENT	$89.74_{\pm1.09}$	$85.19_{\pm1.28}$	$85.19_{\pm1.28}$	$89.35_{\pm1.11}$	$89.22_{\pm1.12}$	$89.09_{\pm1.12}$	$77.40_{\pm1.51}$	$77.40_{\pm1.51}$	$77.40_{\pm1.51}$
SHUTTLE-SMALL	$99.17_{\pm0.21}$	$99.07_{\pm0.22}$	$99.07_{\pm0.22}$	$99.28_{\pm0.19}$	$99.17_{\pm0.21}$	$99.17_{\pm0.21}$	$99.22_{\pm0.20}$	$98.04_{\pm0.32}$	$98.04_{\pm0.32}$
SOYBEAN-LARGE	$85.54_{\pm1.79}$	$87.68_{\pm1.77}$	$86.07_{\pm2.37}$	$84.29_{\pm1.25}$	$84.64_{\pm1.34}$	$86.61_{\pm0.80}$	$50.54_{\pm1.61}$	$50.18_{\pm1.75}$	$48.21_{\pm2.43}$
VEHICLE	$62.72_{\pm1.69}$	$57.28_{\pm1.25}$	$57.51_{\pm1.38}$	$64.85_{\pm1.29}$	$62.49_{\pm1.28}$	$62.60_{\pm1.44}$	$49.94_{\pm0.91}$	$44.73_{\pm1.94}$	$44.73_{\pm1.94}$
VOTE	$94.71_{\pm0.86}$	$90.80_{\pm1.54}$	$91.03_{\pm1.52}$	$94.94_{\pm0.86}$	$95.40_{\pm0.51}$	$95.17_{\pm0.67}$	$95.17_{\pm0.76}$	$95.63_{\pm0.92}$	$95.17_{\pm0.76}$
WAVEFORM-21	$73.34_{\pm0.64}$	$73.64_{\pm0.64}$	$73.64_{\pm0.64}$	$72.26_{\pm0.65}$	$72.28_{\pm0.65}$	$72.26_{\pm0.65}$	$64.38_{\pm0.70}$	$55.85_{\pm0.72}$	$55.85_{\pm0.72}$

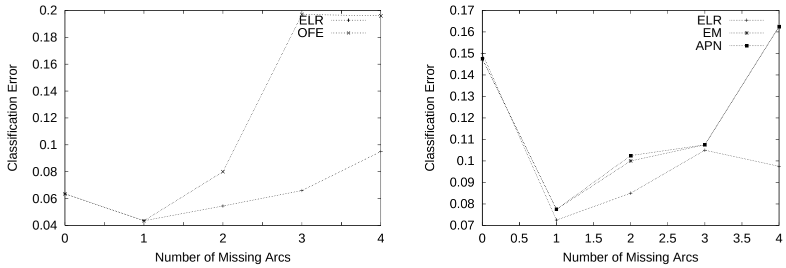

Figure 8. "Correctness of Structure": Comparing ELR to OFE, on increasingly incorrect structures for (a) Complete Data; (b) Incomplete Data;

We next tried to learn the parameters for a TAN structure. Recall the standard TAN-learning algorithm uses the mutual information between each pair of attributes, conditioned on the class variable. This is straightforward to compute 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. Figures 7(a) and 7(b) compare TAN+ELR to TAN+APN and to TAN+EM. We see that these systems are roughly equivalent: while TAN+ELR appears slightly better than TAN+EM, this is not significant (only at $p < 0.25$); similarly there is no significant difference between TAN+ELR and TAN+APN. Finally, we compared NB+ELR to TAN+ELR (Figure 7(c)), but found no significant difference here either.

Table III presents all of our empirical results related to missing data.

We also compared these parameter learners on 20 other UCIrvine datasets that are already missing datapoints. Our results here were consistent: NB+ELR was significantly better than NB+EM and NB+APN — NB+ELR $\Leftarrow_{(p<0.0056)}$ NB+EM, NB+ELR $\Leftarrow_{(p<0.026)}$ NB+APN — but there was no statistical separation difference between TAN+ELR and either TAN+EM or TAN+APN — TAN+ELR $\leftarrow_{(p<0.083)}$ TAN+EM, TAN+ELR $\leftarrow_{(p<0.078)}$ TAN+APN. We provide further details in (Gre04, #MissingData).

5.1.4. *"Correctness of Structure" Study*

The NaïveBayes-assumption, that the attributes are independent given the classification variable, is typically incorrect. This is known to handicap the NaïveBayes classifier in the standard OFE situation; see above and (DP96).

We saw above that ELR is more robust than OFE, in that it is not as handicapped by an incorrect structure. We designed the following simple experiment to empirically investigate this claim.

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) we made NaïveBayes-assumptions and set[12]

$$P(+c) = 0.9 \qquad P(+e_i \,|\, +c) = 0.2 \qquad P(+e_i \,|\, -c) = 0.8 \tag{14}$$

and our queries were all complete; *i.e.*, each instance of the form $\mathbf{E} = \langle \pm e_1, \pm e_2, \ldots, \pm e_k \rangle$.

We then used OFE (resp., ELR) to learn the parameters for the NaïveBayes structure 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 ELR did quite well, efficiently converging to the optimal classification error.

We then tried to learn the CPtables for this NaïveBayes structure, but for distributions that were *not* consistent with this structure. In particular, we formed the m-th distribution P_m by asserting that $E_1 \equiv E_2 \equiv \ldots \equiv E_m$ (*i.e.*, $P(+e_i \,|\, +e_1) = 1.0$, $P(+e_i \,|\, -e_1) = 0.0$ for each $i = 2..m$) in addition to Equation 14. 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 8(a) shows the results, for $k = 5$, based on 400 instances. As predicted, ELR can produce reasonably accurate CPtables here, even for increasingly wrong structures. However, OFE does progressively worse.

"Correctness of Structure", Incomplete Data: We next degraded this training data by randomly removing the value of each attribute, within each instance, with probability 0.5. Figure 8(b) compares ELR with the standard systems APN and EM; again we see that ELR is more accurate, in each case.

5.2. Model Approximates the Truth ($G \approx T$)

The previous section considered learners that were constrained to consider only some limited class of structures, such as NB or TAN. Other learners are allowed to first learn an arbitrary BN structure — seeking one that matches the underlying distribution — before learning the parameters of that structure, using ELR or OFE, etc. There are a number of algorithms for learning these BN structures, each of which will typically produce a structure that is close to correct. This paper considers the PowerConstructor system (CGK02; CG99), which uses mutual information tests to construct BN-structures from complete tuples. This algorithm is guaranteed to converge to the correct belief net structure, given enough data (and some other relatively benign assumptions). We will refer to the resulting PowerConstructor-produced structure as a "General Belief Net", or GBN; see Figure 2(c). This section explores the effectiveness of having such learned structures.

For each of the 25 datasets, we first used PowerConstructor to produce a structure for the given dataset, given all available (non–hold-out) data;

[12] When dealing with binary variables, we let "$+c$" represent $c =$ True, and "$-c$" represent $c =$ False.

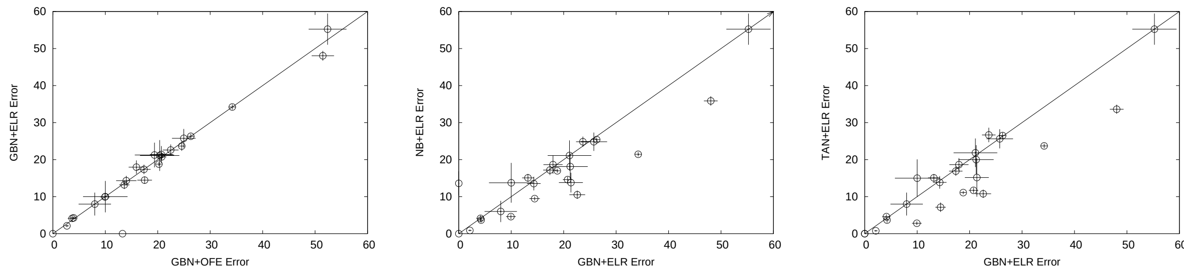

Figure 9. Comparing (a) `GBN+ELR` vs `GBN+OFE`; (b) `GBN+ELR` vs `NB+ELR`; (c) `GBN+ELR` vs `TAN+ELR`

we then asked `ELR` (resp., `OFE`) to find best parameters for this (presumably near optimal) structure using the same non-hold-out data, and observed how well the resulting system performs on the held-out data.

Section 5.2.1 compares `GBN+ELR` to `GBN+OFE`; Section 5.2.2 compares `GBN+ELR` to simpler models instantiated using `ELR`; and Section 5.2.3 compares the `OFE`-instantiation of `GBN` to `ELR`-instantiations of simpler models. Section 5.2.4 investigates different algorithms for learning parameters (for these `GBN` structures) from *incomplete* data.

5.2.1. `GBN+ELR` *vs* `GBN+OFE`

Figure 9(a) shows that `GBN+ELR` is only insignificantly better than `GBN+OFE`: `GBN+ELR` $\leftarrow_{(p<0.2)}$ `GBN+OFE`. Hence, when considering structures that match the underlying distribution, there appears to be little difference between `OFE` and `ELR`.

5.2.2. `GBN+ELR` *vs* `NB+ELR`, `TAN+ELR`

The main purpose of our studies was to see how *y*+`ELR` compares to *y*+`OFE` (and *y*+`EM`/`OFE`), for various classes of commonly-used structures *y*. As a side issue, we also considered some cross-structure comparisons. In particular, given that PowerConstructor had no prior constraints on the structures it can produce, it has the potential of producing classifiers superior to the ones produced by the constrained `NB` or `TAN` systems. However, when we compared `GBN+ELR` to `NB+ELR` (Figure 9(b)), and to `TAN+ELR` (Figure 9(c)), we found that the simpler structures actually produced *better* classifiers than `GBN` did — `NB+ELR` $\Leftarrow_{(p<0.01)}$ `GBN+ELR` and `TAN+ELR` $\Leftarrow_{(p<0.008)}$ `GBN+ELR`.

There are several possible reasons for this. First, the optimization task for `GBN` (unlike the ones for `NB` and `TAN`) is not convex, meaning it could have local, non-global maxima (WGR$^+$03). Of course, the fact that `GBN+ELR` is comparable to `GBN+OFE` shows that this is not a major issue.

Another possibility is that the `GBN` structure might not be good for the classification task: PowerConstructor is seeking a good model of the underlying distribution, but might fail. (Recall its guarantee is asymptotic, and we

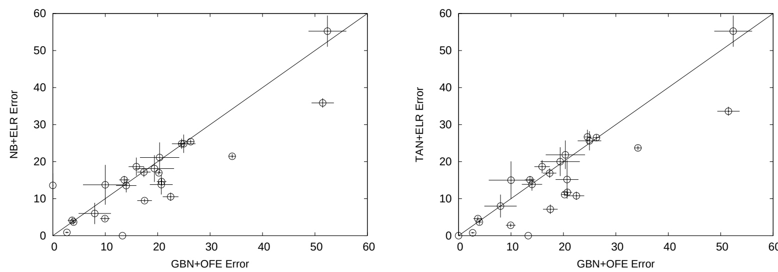

Figure 10. Comparing (a) GBN+OFE vs NB+ELR; (b) GBN+OFE vs TAN+ELR

have only a finite sample). Moreover, even if it obtained a good approximation to the underlying distribution, this might not produce a good classifier; see the arguments presented in Section 2. (As another illustration, imagine the class node C depended on the variables, $\{E_i\}$. POWERCONSTRUCTOR would be happy returning a structure that appeared to match the distribution, even if that structure separated C from many of these relevant E_i's. This cannot happen with either NB or TAN, as these structure connect every variable to C. (Of course, as our goal is to compare y+ELR to y+OFE over commonly used classes y, it does not really matter whether POWERCONSTRUCTOR provided a good structure y or not.)

5.2.3. GBN+OFE *vs* NB+ELR, TAN+ELR

One approach to learning a good belief net (classifier) is to first find a good structure, then instantiate this structure using the trivial OFE algorithm. The first step can be hard — *e.g.*, NP-hard if seeking the structure that maximizes the BIC score (CGH94).[13] Another approach, suggested by our analysis, is to use a simple structure, such as NB or TAN, but then spend resources finding the best parameters, using ELR.

We therefore compared GBN+OFE to NB+ELR (Figure 10) and found NB+ELR to be significantly better: NB+ELR $\Leftarrow_{(p<0.03)}$ GBN+OFE. Moreover, TAN+ELR is yet stronger: TAN+ELR $\Leftarrow_{(p<0.008)}$ GBN+OFE.

Table IV presents a succinct summary of the results on the UCI data, over all 25 datasets. (Note this repeats many of the results from the previous sections.) The rows and columns are ordered based on our expectation that most of the entries would be $\Uparrow$'s. (We see some exceptions, but only when we cross structure classes; see above discussion.)

[13] This BIC is a generative measure. We suspect finding the best "discriminative structure" would be as difficult. See also the iterative methods used in (GD04) for this task.

Table IV. Comparison of Different Parameter Learners — *Complete* (UCI) Data; all 25 datasets

	GBN+ELR	GBN+OFE	TAN+ELR	TAN+OFE	NB+ELR
GBN+OFE	$\uparrow$ (0.2)				
TAN+ELR	$\Leftarrow$ (0.008)	$\Leftarrow$ (0.008)			
TAN+OFE	$\Leftarrow$ (0.04)	$\Leftarrow$ (0.02)	$\uparrow$ (0.12)		
NB+ELR	$\Leftarrow$ (0.01)	$\Leftarrow$ (0.03)	$\uparrow$ (0.2)	$\uparrow$ (0.45)	
NB+OFE	$\uparrow$ (0.36)	$\uparrow$ (0.46)	$\Uparrow$ (0.006)	$\Uparrow$ (0.002)	$\Uparrow$ (0.005)

Legend: Each $\langle i, j\rangle$ entry consists of both an arrow that points to the superior learner (using a double arrow $\Uparrow$ or $\Leftarrow$ if this is significant, and a single arrow $\uparrow$ or $\leftarrow$ otherwise); and the associated p-value in parentheses.

5.2.4. GBN+*x vs other classifiers, with Incomplete data*

This section investigates the effectiveness of learning the parameters for GBN structures, from *incomplete* training data. As POWERCONSTRUCTOR is designed for complete data, we actually built each of the structures using complete data. We did this once, using all of the available data.[14]

To produce the data used for learning and evaluating the parameters, we then removed the values of each evidence attribute for each tuple, with probability 0.25 — so again we are dealing with MCAR data (LR87).

The overall results appear in Table V, where again we expected the majority of the entries to be $\Uparrow$'s. Most importantly, for each class x, we see that x+ELR is sometimes significantly better than x+APN and x+EM, and it is never significantly worse. The fact that both TAN+x and NB+ELR appear uniformly better than GBN+y, is consistent with the case for complete data; see Section 5.2.2.

5.3. MODEL IS MORE COMPLEX THAN TRUTH ($G > T$)

Section 5.1 focused on the common situation where G (the BN-structure being instantiated) is presumedly *simpler* than the "truth" — *e.g.*, we used naïve-bayes when there probably were dependencies between the attributes. This section considers the opposite situation, where we allow the model "more degrees of freedom" than the truth. As this is atypical, we could only consider artificial data.

In our first experiment, we attempt to learn the parameters for a naïve-bayes model, when the truth is $C \equiv E_1$ — *i.e.*, the other attributes $E_2, \ldots, E_k$ are each irrelevant. We focus on $k = 6$ and $k = 7$ attributes, where all variables

[14] As our goal was only to compare the effectiveness of the parameter-learners on reasonable structures, the source of these structures is irrelevant, and in particular, it does not matter that the structure was generated from all the data.

Table V. Comparison of Different Parameter Learners — *InComplete* (UCI) Data; all 25 datasets

	GBN+ELR	GBN+APN	GBN+EM	TAN+ELR	TAN+APN	TAN+EM	NB+ELR	NB+APN
GBN+APN	$\Uparrow$ (0.05)							
GBN+EM	$\uparrow$ (0.09)	$\uparrow$ (0.25)						
TAN+ELR	$\Leftarrow$ (0.015)	$\Leftarrow$ (0.007)	$\Leftarrow$ (0.012)					
TAN+APN	$\Leftarrow$ (0.01)	$\Leftarrow$ (0.005)	$\Leftarrow$ (0.009)	$\uparrow$ (0.32)				
TAN+EM	$\Leftarrow$ (0.015)	$\Leftarrow$ (0.006)	$\Leftarrow$ (0.01)	$\uparrow$ (0.25)	$\uparrow$ (0.5)			
NB+ELR	$\Leftarrow$ (0.02)	$\Leftarrow$ (0.01)	$\Leftarrow$ (0.015)	$\uparrow$ (0.4)	$\uparrow$ (0.32)	$\uparrow$ (0.3)		
NB+APN	$\uparrow$ (0.08)	$\uparrow$ (0.06)	$\Uparrow$ (0.04)	$\uparrow$ (0.07)	$\uparrow$ (0.06)	$\Uparrow$ (0.04)	$\Uparrow$ (0.025)	
NB+EM	$\uparrow$ (0.06)	$\Uparrow$ (0.05)	$\Uparrow$ (0.04)	$\uparrow$ (0.055)	$\Uparrow$ (0.05)	$\Uparrow$ (0.035)	$\Uparrow$ (0.015)	$\uparrow$ (0.2)

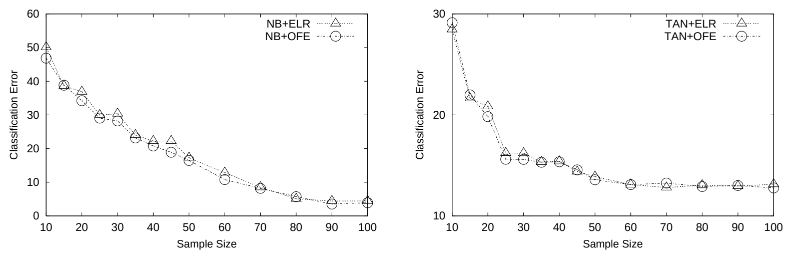

Figure 11. $G > T$ situations, complete data. (a) Model is NB; Truth is $C \equiv E_1$; (b) Model is TAN; Truth is NaïveBayes. (Each point is averaged over 10 runs)

are binary. When the data is complete, we used first OFE and then ELR to instantiate the parameters of a given NaïveBayes model. Figure 11(a) shows the learning curve as we increase the sample size, over 10 different runs. (Each run used its own training sample.) We see that NB+OFE is consistently slightly better than NB+ELR: averaged over all of the runs, this is significant at $p < 0.002$.

We also weakened the $C \equiv E_1$ condition, to simply require C be highly correlated with E_1. Using the same set-up show above, when the correlation is 0.96, we found NB+OFE $\Leftarrow_{(p<0.001)}$ NB+ELR. When the correlation is 0.80, the dominance is even more: NB+OFE $\Leftarrow_{(p<0.0001)}$ NB+ELR.

The second experiment "reverses" the situations shown in Section 5.1.4 . Here, the truth corresponds to a naïve-bayes structure (with no dependencies between the evidence E_i variables, conditioned on the class variable), but we attempt to find the parameters for a "P_m-based structure" — *i.e.*, a TAN structure that links $E_1 \equiv E_2 \equiv \ldots \equiv E_m$. These results appear in Figure 11(b), again this is averaged over 10 runs. (This difference is not significant.)

We next considered the same two situations, but in the *incomplete* data case. In particular, here we blocked a value of any entry with probability 0.2.

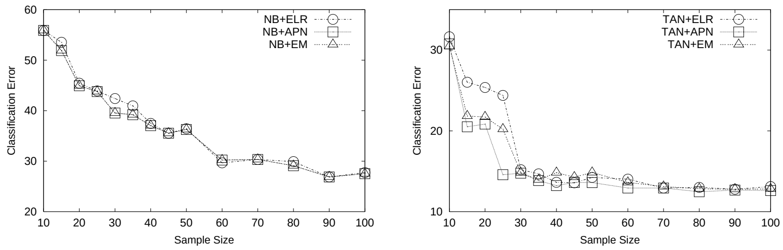

Figure 12. $G > T$ situations, *incomplete* data. (a) Model is NB; Truth is $C \equiv E_1$; (b) Model is TAN; Truth is NaïveBayes. (Each point is averaged over 10 runs.)

The results, shown in Figure 12, show that the generative measures (NB+APN and NB+EM) dominated the discriminative NB+ELR: NB+APN $\Leftarrow_{(p<0.02)}$ NB+ELR and NB+EM $\Leftarrow_{(p<0.015)}$ NB+ELR. (Moreover, NB+EM $\Leftarrow_{(p<0.025)}$ NB+APN.) The generative approach is also superior in the other sitation (Figure 12(b)): TAN+APN $\Leftarrow_{(p<0.025)}$ TAN+ELR, and TAN+EM $\Leftarrow_{(p<0.05)}$ TAN+ELR.

In a nutshell, we observed that discriminative ELR learning typically did worse than the generative learners in this "model is more complex than truth" situation, when dealing with either complete or incomplete data.

5.4. Discussion

This section has presented a large number of empirical results, all in the context of producing a good belief-net based classifiers for a fixed structure. The main take-home messages are...

- In the unusual situation where the user is forced to use a model G that is more complex than the truth T, it is better to use the generative learners (OFE, APN, EM) — Section 5.3 .[15] However...
- In essentially all other complete-data situations, the discriminative learner ELR is at least as good, and often superior, to OFE. See in particular the x+ELR vs x+OFE entries in Table IV.
- We see similar results in the *incomplete* data case; here, the discriminative learner ELR is at least as good as, and often superior to. APN and EM. See Table V.
- While we typically found more expressive models produced better classifiers (*i.e.*, for each parameter-learner z, GBN+z was better than TAN+z, and TAN+z was better than NB+z), this was not universal; see discussion in Section 5.2.2.

[15] Here, both types of learner — both generative and discriminative — are learning classifiers that *can* represent the optimal parameters, by simply setting the CPtable entries of the extra arcs to be uniform, which effectively ignores those arcs. As discriminative learners are less constrained, they can overfit, especially as the structures become more complex.

Why ELR **Works Well:** We found that ELR worked effectively in many situations, and it was especially advantageous (*i.e.*, typically better than the alternative ways to instantiate parameters) when the BN-structure was *incorrect* — *i.e.*, when 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), TAN (FGG97), etc.) that is not guaranteed to contain the correct structure.

To understand why a bad structure is problematic for OFE, note that when OFE is seeking the parameter $\theta_{d|\mathbf{f}}$, it is constrained to match the *local* empirical distribution, corresponding to $\#(D=d,\mathbf{F}=\mathbf{f})/\#(\mathbf{F}=\mathbf{f})$. Hence, if the given structure G is incorrect, the resulting instantiated belief net need not be a good model of the true tuple distribution, and so may return incorrect values for the queries. By contrast, the ELR 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.)

Computational Efficiency: ELR typically required a handful of iterations to converge for the small datasets, and dozens of iterations for the larger ones. APN and EM typically used slightly more iterations. Our current ELR implementation is in unoptimized JAVA code. Its time per iteration varied, from around 0.5 seconds per iteration for the smaller datasets through a few minutes for larger datasets on a PentiumIII-800MHz. 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. (Of course, OFE only applies to complete data situations, while ELR applies in general.) We are currently investigating whether there could be more efficient algorithms for our task; see Section 7.1.

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 ELR.

Other Learners: Finally, we compared ELR to various other learning algorithms, including SVMs, over these datasets. We found that ELR was comparable with several other standard learning algorithms, and superior to some, at least over these datasets. See (Gre04, #OtherLearners) for details, which also repeats the results from Friedman *et al.* (FGG97) and Grossman and Domingos (GD04). That webpage also provides other information about the experiments; *e.g.*, presenting the $\widehat{\mathrm{LCL}}(\cdot)$ scores, etc.

6. Related Results

There are a number of other results related to learning belief nets. Much of this 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 13). This is well motivated in the *generative* situation, as these parameter values do optimize the likelihood of the data (CH92).

As noted earlier, however, our goal is different: as we are seeking the optimal *classifier* — *i.e.*, *discriminative 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. We are considering eventual performance systems that will be expected to address a certain range of questions — *e.g.*, about the probability of cancer given gender, age and smoking habits. 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 discriminative learning of belief nets (KMST99; JMJ00; FGG97; CG99; GD04). This research, like their generative counterparts, focused on learning structures; and usually used `OFE` to instantiate the resulting parameters.

As we saw above, when the model is wrong, the `OFE`-based parameters can produce inferior classifiers. Many researchers have employed tricks to improve the parameters; *e.g.*, tractable Bayesian model averaging of TAN (CdM03), and exact model averaging of naïve-bayes (DC02). Our approach is different, as we explicitly seek the parameters of the BN model that maximizing conditional likelihood.

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 "Generalized Probabilistic Descent", which resembles our ELR system by descending along the derivative of the parameters, to maximize the conditional likelihood of the hypothesis (which typically correspond to specific words) given the observations — which they call "Maximimum Mutual Information" criterion. We differ by considering arbitrary structures, and evaluating based on classification error.

Edwards & Lauritzen (EL01) proposed the TM algorithm for maximizing conditional likelihood function, when the corresponding uncondition likelihood function is more easily maximized. They have found the algorithm is a useful tool in complex CG-regression models, which are the building blocks for graphical chain models, as well as in other situations (Sun02). Their TM algorithm is similar to the EM algorithm as it also alternates between maximization of a function related to the true likelihood function, but differs by being applied to the complete data case and by augmenting the parameters rather than the data.

This relates directly to the large literature on *discriminative learning* in general; see (CS89; Jor95; Rip96). One standard model is Linear Discriminant Analysis (LDA), which typically assumes $P(\mathbf{E}\,|\,C=c)$ is multivariate normal — *i.e.*, $P(\mathbf{E}\,|\,C=c) \sim \mathcal{N}(\mu_c, \Sigma)$ where each μ_c mean can depend on the class $C=c$, but the covariance matrix Σ 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 those parameters. Given these parameters, we can then use Bayes Rule to compute the conditional distribution of $P(C\,|\,\mathbf{E}=\mathbf{e}')$ given new evidence $\mathbf{E}=\mathbf{e}'$.

We can view LDA (like OFE/APN/EM) 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 our ELR is discriminative (aka "diagnostic", "predictive" (Jor95)), as it focuses only on the *conditional* probabilities.

Our results echo the common wisdom obtained by the previous analyses of discriminative systems. In particular, (1) accuracy: discriminative training typically produces more accurate classifiers than generative training; (2) robustness: typically discriminative systems are more robust against incorrect models than generative ones; (3) efficiency: generative can be more efficient than discriminative (compare the efficient OFE with the iterative ELR). Due to the final point, many discriminative learners initialize their parameters based on generative (read "maximum-likelihood") estimates, especially as the latter are often "plug-in parameters" (Rip96). (As mentioned in Section 4, our ELR algorithm incorporates this idea as well.)

The work reported in this paper has significant differences from those earlier analyses. First, we are dealing with a different underlying model, based on *discrete* variables (rather than continuous, say normally distributed, ones), in the context of a *specified belief net structure*, which corresponds to a given set of independency claims. We also describe the inherent computational complexity of this task, produce algorithms specific to our task, and provide empirical studies to demonstrate that our algorithm 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 9 that is based on two different types of samples — one over tuples, to estimate $P(C\,|\,\mathbf{E})$, and the other over queries, to estimate the probability of seeing each "What is $P(C\,|\,\mathbf{E}=\mathbf{e})$?" query. By contrast, the current paper tries to minimize classification error (Equation 3) by seeking the optimal "conditional likelihood" score (Equation 4), wrt a single sample of labeled instances. Moreover, our current paper includes new theoretical results, a different algorithm, and completely new empirical data.

7. Conclusions

7.1. Future Work

Section 3 notes that, in some situations (a specified class of structures, when given complete data), interior point methods can find the parameters that optimize conditional likelihood, in polynomial time. Of course, it is not clear whether these $\mathrm{LCL}(\cdot)$-optimal parameters will optimize error (Equation 3), nor that the algorithm will necessarily be more efficient than `ELR`. We therefore plan to investigate these methods.

This paper explores the challenges of finding 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; see (GD04).

There are now several other classes of graphical models, such as Conditional Random Fields (LMP01), that may be better adapted to optimizing conditional likelihood. (*E.g.*, as they use undirected arcs, they require only a single normalizing division, rather than one per CPtable row.) While this paper has focused on Belief Nets (as it is typically easier to acquire meaningful structures here, both because they allow users to express their prior knowledge, and because there are a number of algorithms for learning belief net structures), we plan to investigate these other models as well.

So far, the goal is classification accuracy. This measure is not as useful when the dataset is imbalanced (*i.e.*, many more instances of one class than

Table VI. Summary of Known Complexity Results

	Complete Data	Incomplete Data
Likelihood	in P: (OFE)	Unknown (EM, APN)
Conditional Likelihood	in P for simple structures (WGR$^+$03) Unknown in general	NP-hard (Theorem 2)

another) or when different misclassifications have different penalties. Here, it is common to seek the classifier (here, set of parameters) that maximize the area-under-ROC-curve (AUC) measure (HT01). We plan to explore this approach.

7.2. Contributions

This paper overviews the task of discriminative learning of belief net parameters for general BN-structures. We first describe this task, and discuss how it extends that standard logistic regression process by applying to arbitrary structures, not just naïve-bayes — see Equation 2. It is well known that discriminative learning can converge to a classifier superior to one learned generatively (NJ01). Our formal analyses show that, in general, discriminative learners can converge to a classifier optimizing conditional likelihood at essentially the same $O(\cdot)$ sample rate (ignoring polylog terms) as a generative classifier that is optimizing likelihood. (This differs from the Ng & Jordan (NJ01) result, which compared only the simplest form, NaïveBayes vs logistic regression, and dealt with error itself.) We also found that the *computational* complexities of these two tasks also appear fairly comparable; see Table VI.

We next present an algorithm ELR for our task, and show that ELR works effectively over a variety of situations: when dealing with structures that range from trivial (NB), through less-trivial (TAN), to complex (ones learned by PowerConstructor). We also show that ELR works well when given *incomplete* training data. Our empirical evidence suggests that ELR can be inferior to the standard generative models only in the unusual situation where the model is more complex than the truth. In essentially every other situation, however, we see that ELR is at least as good, and often better, than the other contenders. We also include a short study to explain why ELR can work effectively, showing that it typically works better than generative methods when dealing with models that are less complicated than the true distribution, which is a very common situation.

While statisticians are quite familiar with the idea of discriminative learning (*e.g.*, logistic regression), this idea, in the context of belief nets, is only beginning to make in-roads into the general AI community. We hope this paper will help further introduce these ideas to this community, and demonstrate that these algorithms should be used here, as they can work very effectively.

For more information, including all of the data used for the experiments, see (Gre04).

ACKNOWLEDGEMENTS

We thank Corrine Cheng, Tom Dietterich, Adam Grove, Peter Hooper, John Lafferty, Chris O'Brien, Dale Schuurmans, Lyle Ungar and the anonymous reviewers for their many helpful suggestions. We also thank Jie Cheng for allowing us to use his POWERCONSTRUCTOR system for our GBN studies, and Thorsten Joachims for letting us use his SVM-Light system. RG and WZ were partially funded by NSERC; RG was also funded by the Alberta Ingenuity Centre for Machine Learning; and WZ, by Syncrude.

References

N. Abe, J. Takeuchi, and M. Warmuth. Polynomial learnability of probablistic concepts with respect to the Kullback-Leibler divergence. In *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 277–289. Morgan Kaufmann, 1991.

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

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

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.

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

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Jesús Cerquides and Ramon López de Mántaras. Tractable Bayesian learning of tree augmented naïve Bayes models. In *ICML-03*, pages 75–82, 2003.

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.

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

Jie Cheng, Russell Greiner, and Jonathan Kelly. Learning Bayesian networks from data: an information-theory based approach. *Artificial Intelligence*, 137:43–90, 2002.

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

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

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

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

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

Sanjoy Dasgupta. The sample complexity of learning fixed-structure Bayesian networks. *Machine Learning*, 29:165–180, 1997.

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

Denver Dash and Gregory Cooper. Exact model averaging with naïve Bayesian classifiers. In *ICML-02*, pages 91–98, 2002.

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

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. (with discussion). *Journal of the Royal Statistics Society, Series B*, 39:1–38, 1977.

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.

David Edwards and Steffen Lauritzen. The TM algorithm for maximising a conditional likelihood function. *Biometrika*, 88:961–972, 2001.

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

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.

D. Grossman and P Domingos. Learning Bayesian network classifiers by maximizing conditional likelihood. In *ICML2004*, 2004.

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

2004. http://www.cs.ualberta.ca/∼greiner/ELR.

Clark Glymour, Richard Scheines, Peter Spirtes, and Kevin Kelly. *Discovering Causal Structure*. Academic Press, Inc., London, 1987.

Russell Greiner and Wei Zhou. Structural extension to logistic regression: Discriminant parameter learning of belief net classifiers. In *Proceedings of the Eighteenth Annual National Conference on Artificial Intelligence (AAAI-02)*, pages 167–173, Edmonton, August 2002.

M. Hagan, H. Demuth, and M. Beale. *Neural Network Design*. PWS Publishing, Boston, MA, 1996.

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

David J. Hand and Robert J. Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning Journal*, 45:171–186, 2001.

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

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

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

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

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.

S. Lauritzen. The em algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19:191–201, 1995.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML2001*, 2001.

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

Tom Minka. Algorithms for maximum-likelihood logistic regression. Technical report, CMU CALD, 2001. http://www.stat.cmu.edu/∼minka/papers/logreg/minka-logreg.pdf.

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

P. McCullagh and J. Nelder. *Generalized Linear Models*. Chapman and Hall, 1989.

C. Nadeau and Y. Bengio. Inference for the generalization error. *Machine Learning*, 52:239–281, 2003.

A. Y. Ng and M. I. Jordan. On discriminative versus generative classifiers: A comparison of logistic regression and naive Bayes. In *Advances in Neural Information Processing Systems 14*, 2001.

Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Methods in Convex Programming*. Society for Industrial and Applied Mathematics, 1994.

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

William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C*. Cambridge, 2002. http://www.nr.com/.

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

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.

Bin Shen, Xiaoyuan Su, Russell Greiner, Petr Musilek, and Corrine Cheng. Discriminative parameter learning of general Bayesian network classifiers. In *Proceedings of the Fifteenth IEEE International Conference on Tools with Artificial Intelligence (ICTAI-03)*, Sacramento, November 2003.

Rolf Sundberg. The convergence rate of the TM algorithm of Edwards and Lauritzen. *Biometrika*, 89:478–483, 2002.

Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

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

H. Wettig, P. Grünwald, T. Roos, P. Myllymäki, and H. Tirri. When discriminative learning of Bayesian network parameters is easy. In *IJCAI2003*, pages 491–496, 2003.

Wei Zhou. Discriminative learning of bayesian net parameters. Master's thesis, Dept of COmputing Science, University of Alberta, 2002.

Appendix

A. Proofs

Proof of Theorem 1: As the set $\mathcal{BN}_{\Theta\succeq\gamma}(G)$ is uncountably infinite, we cannot simply apply the standard techniques for PAC-learning a finite hypothesis set. We can, however, partition this uncountable space into a finite number $L = L(K,\gamma,\varepsilon)$ of sets, such that any two BNs within a partition have similar conditional log-likelihood scores. We can then, in essense, simultaneously estimate the scores of all members of $\mathcal{BN}_{\Theta\succeq\gamma}(G)$ if we collect enough query instances to estimate the score for one representative of each partition.

Now for the details: We prove below that, if the CPtables for two BNs $\Theta^{(1)}, \Theta^{(2)} \in \mathcal{BN}_{\Theta\succeq\gamma}(G)$ have similar CPtables $\Theta^{(1)} = \{\theta^{(1)}_{d_i|\mathbf{f}_i}\}_i$ and $\Theta^{(2)} = \{\theta^{(2)}_{d_i|\mathbf{f}_i}\}_i$, then they will have similar LCL-scores wrt any query; *i.e.*,

$$\text{if} \quad \left|\theta^{(1)}_{d_i|\mathbf{f}_i} - \theta^{(2)}_{d_i|\mathbf{f}_i}\right| \leq \frac{\gamma\varepsilon}{6K} \quad \text{then} \quad \forall c, \mathbf{e}\ |\ln(P_{\Theta^{(1)}}(\,c\,|\,\mathbf{e}\,)) - \ln(P_{\Theta^{(2)}}(\,c\,|\,\mathbf{e}\,))| \leq \frac{\varepsilon}{6}\,. \tag{15}$$

This of course implies the same bound on the difference between their overall LCL-scores

$$|\mathrm{LCL}_k(\,\Theta^{(1)}\,) - \mathrm{LCL}_k(\,\Theta^{(2)}\,)| \quad \leq \quad \frac{\varepsilon}{6}$$

for any distribution $\mathrm{LCL}_k(\,\cdot\,)$ — both for the "true" query distribution $\mathrm{LCL}(\,\cdot\,)$, and for the distribution associated with any empirical sample $\widehat{\mathrm{LCL}}(\,\cdot\,)$.

We therefore partition the $\mathcal{BN}_{\Theta\succeq\gamma}(G)$ space into $L = (\frac{6K}{\gamma\varepsilon})^K$ disjoint sets (where any two BNs from any partition will have similar CPtable values), then define the set $R = \{\Theta_i\}_i$ to contain one representative from each partition. We prove below that a sample S of size

$$M\left(\frac{\varepsilon}{6}, \frac{\delta}{L}\right) \quad = \quad 2\left(\frac{3N\log\gamma}{\varepsilon}\right)^2 \ln\frac{2L}{\delta} \tag{16}$$

is sufficient to estimate each of these single representatives to within $\varepsilon/6$ of correct, with probability of error at most δ/L; *i.e.*, such that, for each i,

$$P\left[\,\left|\widehat{\mathrm{LCL}}^{(S)}(\,\Theta_i\,) - \mathrm{LCL}(\,B_i\,)\right| > \frac{\varepsilon}{6}\,\right] \quad < \quad \frac{\delta}{L}\,.$$

As there are L representatives, we have a total probability of at most $L\frac{\delta}{L} = \delta$ that *any* of the representative's scores are mis-estimated by more than $\varepsilon/6$.

This means we have, in effect, estimated the scores on *any* $\Theta \in \mathcal{BN}_{\Theta\succeq\gamma}(G)$ to within $\varepsilon/2$: For any $\Theta \in \mathcal{BN}_{\Theta\succeq\gamma}(G)$, let $\Theta' \in R$ be the representative in Θs partition. Observe

$$\begin{array}{rcccccccc} |\widehat{\mathrm{LCL}}(\,\Theta\,) - \mathrm{LCL}(\,\Theta\,)| & \leq & |\widehat{\mathrm{LCL}}(\,\Theta\,) - \widehat{\mathrm{LCL}}(\,\Theta'\,)| & + & |\widehat{\mathrm{LCL}}(\,\Theta'\,) - \mathrm{LCL}(\,\Theta'\,)| & + & |\mathrm{LCL}(\,\Theta'\,) - \mathrm{LCL}(\,\Theta\,)| \\ & \leq & \varepsilon/6 & + & \varepsilon/6 & + & \varepsilon/6 \\ & = & \varepsilon/2\,. & & & & \end{array}$$

This means, in particular, that our estimate of the scores of both $\widehat{\Theta}$ and Θ^* are within $\varepsilon/2$, and so

$$\begin{array}{rcccccccc} \mathrm{LCL}(\,\widehat{\Theta}\,) - \mathrm{LCL}(\,\Theta^*\,) & \leq & |\mathrm{LCL}(\,\widehat{\Theta}\,) - \widehat{\mathrm{LCL}}(\,\widehat{\Theta}\,)| & + & \widehat{\mathrm{LCL}}(\,\widehat{\Theta}\,) - \widehat{\mathrm{LCL}}(\,\Theta^*\,) & + & |\widehat{\mathrm{LCL}}(\,\Theta^*\,) - \mathrm{LCL}(\,\Theta^*\,)| \\ & \leq & \varepsilon/2 & + & 0 & + & \varepsilon/2 \end{array}$$

To complete the proof, we need only prove Equations 15 and 16. For Equation 15: Consider the sequence of BNs $\Theta_0, \Theta_1, \ldots, \Theta_K$ where the first i of Θ_i's CPtables come from $\Theta^{(1)}$, and the remaining from $\Theta^{(2)}$ — *i.e.*,

$$\Theta_i \quad \sim \quad \{\theta^{(1)}_{d_1|\mathbf{f}_1}, \ldots, \theta^{(1)}_{d_i|\mathbf{f}_i}, \theta^{(2)}_{d_{i+1}|\mathbf{f}_{i+1}}, \ldots, \theta^{(2)}_{d_K|\mathbf{f}_K}\} .$$

Now observe

$$|\ln(P_{\Theta^{(1)}}(c\,|\,\mathbf{e})) - \ln(P_{\Theta^{(2)}}(c\,|\,\mathbf{e}))| \quad \leq \quad \sum_{i=1}^{K} |\ln(P_{\Theta_i}(c\,|\,\mathbf{e})) - \ln(P_{\Theta_{i-1}}(c\,|\,\mathbf{e}))| \,,$$

and each $|\ln(P_{\Theta_i}(c\,|\,\mathbf{e})) - \ln(P_{\Theta_{i-1}}(c\,|\,\mathbf{e}))|$ is based on changing a single CPtable entry. We therefore need only show $|\ln(P_{\Theta_i}(c\,|\,\mathbf{e})) - \ln(P_{\Theta_{i-1}}(c\,|\,\mathbf{e}))| \leq \frac{\varepsilon}{6K}$. For any value of $z = \theta_{d_i|\mathbf{f}_i}$, let $f(z) = \ln(P_{\Theta[z]}(c\,|\,\mathbf{e}))$, where $\Theta[z]$ be the BN whose first $i-1$ CPtable entries come from $\Theta^{(1)}$, whose final $K-i-1$ entries come from $\Theta^{(2)}$, and whose i^{th} CPtable entries is z; hence $f(\theta^{(1)}_{d_i|\mathbf{f}_i}) = \ln(P_{\Theta_i}(c\,|\,\mathbf{e}))$, and $f(\theta^{(2)}_{d_i|\mathbf{f}_i}) = \ln(P_{\Theta_{i+1}}(c\,|\,\mathbf{e}))$. As this function is continuous, we know that

$$|f(a) - f(b)| \quad = \quad \frac{\partial f(z)}{\partial z}[b-a]$$

for some $z \in [a,b]$. As $f(z) = \ln(P_{\Theta[z]}(c, \mathbf{e})) - \ln(P_{\Theta[z]}(\mathbf{e}))$, we see that

$$\begin{aligned} \frac{\partial f(z)}{\partial z} &= \frac{1}{P_{\Theta[z]}(c,\mathbf{e})} P_{\Theta[z]}(c, \mathbf{e}\,|\,d_i, \mathbf{f}_i) \times P_{\Theta[z]}(\mathbf{f}_i) \;-\; \frac{1}{P_{\Theta[z]}(\mathbf{e})} P_{\Theta[z]}(\mathbf{e}\,|\,d_i, \mathbf{f}_i) \times P_{\Theta[z]}(\mathbf{f}_i) \\ &= \frac{1}{z}[P_{\Theta[z]}(d_i, \mathbf{f}_i\,|\,c, \mathbf{e}) - P_{\Theta[z]}(d_i, \mathbf{f}_i\,|\,\mathbf{e})] \end{aligned}$$

which means that $|\frac{\partial f(z)}{\partial z}| \leq 1/z \leq 1/\gamma$. (The second inequality follows from the assumption that we are only considering $\Theta \in \mathcal{BN}_{\Theta \succeq \gamma}(G)$.) Hence,

$$\begin{aligned} |\ln(P_{\Theta_{i+1}}(c\,|\,\mathbf{e})) - \ln(P_{\Theta_i}(c\,|\,\mathbf{e}))| &= |f(\theta^{(2)}_{d_i|\mathbf{f}_i}) - f(\theta^{(1)}_{d_i|\mathbf{f}_i})| \\ &\leq \frac{1}{\gamma} \times |\theta^{(2)}_{d_i|\mathbf{f}_i} - \theta^{(1)}_{d_i|\mathbf{f}_i}| \quad \leq \quad \frac{1}{\gamma} \times \frac{\gamma\varepsilon}{6K} \quad = \quad \frac{\varepsilon}{6K} \,. \end{aligned}$$

To prove Equation 16: Observe first that the probability of any event must be at least the product of N CPtable entries, and hence $P_\Theta(c) \geq \gamma^N$ for any c and any $\Theta \in \mathcal{BN}_{\Theta \succeq \gamma}(G)$. This means the value of $-\ln(P_\Theta(c\,|\,\mathbf{e}))$, and hence $\mathrm{LCL}_{sq}(\Theta)$ for any distribution sq, is between 0 and $-N \ln \gamma$.

As the queries $q = P(c, \mathbf{e})$ are drawn at random from a stationary distribution, we can view the quantity $\ln P_\Theta(q)$ as an iid random value, whose range is $[0, -N\ln\gamma]$ and whose expected value is $\mathrm{LCL}(\Theta)$. Hoeffding's Inequality bounds the chance that the empirical average score after M iid examples (here $\widehat{\mathrm{LCL}}^{(S)}(\Theta)$) will be far away from the true mean $\mathrm{LCL}(\Theta)$:

$$P(|\widehat{\mathrm{LCL}}^{(S)}(\Theta) - \mathrm{LCL}(\Theta)| > \frac{\varepsilon}{6}) \quad < \quad 2\exp\left[-2M((\varepsilon/6)/N\ln\gamma)^2\right] . \quad (17)$$

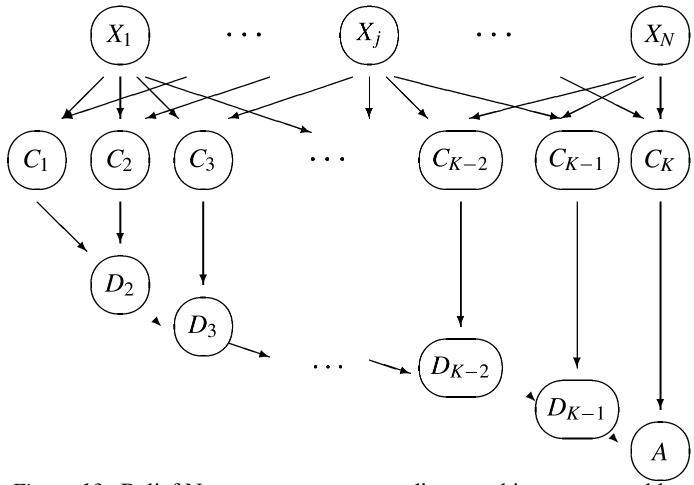

Figure 13. Belief Net structure corresponding to arbitrary SAT problem (Coo90)

Here, we want the right-hand-side to be under δ/L, which requires $M = M(\varepsilon,\delta) = 2\left(\frac{3N\ln\gamma}{\varepsilon}\right)^2 \ln(\frac{2L}{\delta})$. ∎

Proof of Theorem 2: 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 13, 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 $C_1 = x_1 \vee \neg x_2 \vee x_3$ and $C_2 = \neg x_1 \vee \neg x_3 \vee x_4$, then there are links to C_1 from each of X_1, X_2 and X_3, and to 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 C_i variables. We do this using all-but-the-final instances in Table VII. (Note only 3 of the X_i variables are specified in each of these instances; the other $n-3$ X_is are not, nor are any C_js nor D_ks.) 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 $C_1 \equiv x_1 \vee \neg x_2 \vee x_3$. The final instance is just stating that the prior value for A should $P(+a) = 1.0$. The "label" of each instance always corresponds to the single variable A.

We now prove, in particular, that

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

Table VII. Queries used in proof of Theorem 2

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

$\Leftarrow$**:** Just set the CPtable for each C_i to be the disjunction of the associated X_{i1}, X_{i2}, X_{i3} variables (its parents), with the appropriate $\pm$ parity. *E.g.*, using $C_1 \equiv x_1 \vee \neg x_2 \vee x_3$, then C_1's CPtable would be

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	0.0
0	1	1	1.0
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 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

$P(+x_1)$
1.0

; and if *i.e.*, if $X_4 = 0$, then

$P(+x_4)$
0.0

. Note that these CPtable values satify all $k+1$ of the labeled instances.

$\Rightarrow$**:** 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 labeled 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

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

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 3: Below, we will use $P(\chi)$ to refer to $P_\Theta(\chi)$, the value the belief net with parameters Θ will assign to the χ event. In general, for any assignment Z,

$$P(Z) \quad = \quad \sum_{\mathbf{f}'}\sum_{d'} P(Z \,|\, D{=}d', \mathbf{F}{=}\mathbf{f}')\; P(D{=}d' \,|\, \mathbf{F}{=}\mathbf{f}')\; P(\mathbf{F}{=}\mathbf{f}')\;. \qquad (18)$$

As we assume the different CPtable rows are estimated independently, and $\mathbf{F}$ is the set of parents of D, this means

$$\frac{\partial\, P(Z)}{\partial\, \beta_{d|\mathbf{f}}} \quad = \quad \sum_{d'} P(Z \,|\, d', \mathbf{f})\, \frac{\partial\, P(d' \,|\, \mathbf{f})}{\partial\, \beta_{d|\mathbf{f}}} P(\mathbf{f})\;.$$

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

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

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

∎