Discriminative Parameter Learning for Bayesian Networks

Jiang Su JSU@SITE.UOTTAWA.CA
School of Information Technology and Engineering University of Ottawa, K1N 6N5 Canada

Harry Zhang HZHANG@UNB.CA
Faculty of Computer Science, University of New Brunswick, Fredericton, NB, E3B 5A3, Canada

Charles X. Ling CLING@CSD.UWO.CA
Department of Computer Science, The University of Western Ontario, London, Ontario, N6A 5B7, Canada

Stan Matwin STAN@SITE.UOTTAWA.CA
School of Information Technology and Engineering University of Ottawa, K1N 6N5 Canada

Abstract

Bayesian network classifiers have been widely used for classification problems. Given a fixed Bayesian network structure, parameters learning can take two different approaches: generative and discriminative learning. While generative parameter learning is more efficient, discriminative parameter learning is more effective. In this paper, we propose a simple, efficient, and effective discriminative parameter learning method, called *Discriminative Frequency Estimate* (DFE), which learns parameters by discriminatively computing frequencies from data. Empirical studies show that the DFE algorithm integrates the advantages of both generative and discriminative learning: it performs as well as the state-of-the-art discriminative parameter learning method ELR in accuracy, but is significantly more efficient.

1. Introduction

A Bayesian network (BN) (Pearl, 1988) consists of a directed acyclic graph G and a set P of probability distributions, where nodes and arcs in G represent random variables and direct correlations between variables respectively, and P is the set of local distributions for each node. A local distribution is typically specified by a conditional probability table (CPT). Thus, learning Bayesian networks from data has two elements: structure learning and parameter learning.

Bayesian networks are often used for classification problems, in which a learner attempts to construct a classifier from a given set of training instances with class labels. In learning Bayesian network classifiers, parameter learning often uses *Frequency Estimate* (FE), which determines parameters by computing the appropriate frequencies from data. The major advantage of FE is its efficiency: it only needs to count each data point (training instance) once. It is well-known that FE maximizes likelihood and thus is a typical generative learning method.

In classification, however, the objective is to maximize generalization accuracy, rather than likelihood. Thus, discriminative parameter learning that maximizes generalization accuracy or its alternative objective function, conditional likelihood, are more desirable. Unfortunately, there is no closed form for choosing the optimal parameters, because conditional likelihood does not decompose (Friedman et al., 1997). As a consequence, discriminative parameter learning for Bayesian networks often resorts to search methods, such as gradient descent.

Greiner and Zhou (2002) proposed a gradient descent based parameter learning method, called ELR, to discriminatively learn parameters for Bayesian network classifiers, and showed that ELR significantly outperforms the generative learning method FE. However, the application of ELR is limited due to its high computational cost. For example, Grossman and Domingos (2004) observed that ELR is computationally infeasible in structure learning. In fact, how to find an

Appearing in *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finland, 2008. Copyright 2008 by the author(s)/owner(s).

efficient and effective discriminative parameter learning for Bayesian network classifiers is an open question.

In this paper, we propose a simple, efficient, and effective discriminative parameter learning method, called *Discriminative Frequency Estimate* (DFE). Our motivation is to turn the generative parameter learning method FE into a discriminative one by injecting a discriminative element into it. DFE discriminatively computes frequencies from data, and then estimates parameters based on the appropriate frequencies. Our empirical studies show that DFE inherits the advantages of both generative and discriminative learning.

2. Related Work

Greiner and Zhou (2002) showed that discriminative parameter learning for Bayesian networks is equivalent to a logistic regression problem under certain conditions. For many Bayesian network structures, they indicated that the conditional likelihood function may have only one global maximum, and thus can be maximized by local optimization methods. They also proposed a gradient descent based parameter learning method, called ELR. To make ELR work effectively, they modified the basic gradient descent method using FE to initialize parameters and cross tuning to prevent overfitting. Empirical studies showed that ELR significantly outperforms the generative learning approach.

Grossman and Domingos (2004) proposed a discriminative structure learning method for Bayesian network classifiers, and tried to combine discriminative structure learning with discriminative parameter learning. To overcome the efficiency problem of ELR, they reduced the fold of cross tuning, and used a small sample for parameter learning. They observed that the modified ELR still takes two orders of magnitude of learning time longer than FE in their experiments, and the performance of the combination of discriminative structure and parameter learning does not outperform the discriminative structure learning alone. Therefore, they suggested learning a structure by conditional likelihood, and setting parameters by the FE method.

To our knowledge, ELR is the state-of-the-art algorithm for discriminative parameter learning for Bayesian network classifiers. Unfortunately, its computational cost is quite high. In this paper, we propose a discriminative parameter learning algorithm that is as effective as ELR but much more efficient.

3. Frequency Estimate

We use capital letters X for a discrete random variable. The lower-case letters x is used for the value taken by variable X, and x_{ij} refers to the variable X_i taking on its j_{th} value. We use the boldface capital letters $\mathbf{X}$ for a set of variables, and the boldface lower case letters $\mathbf{x}$ for the values of variables in $\mathbf{X}$. The training data D consists of a set of finite number of training instances, and an instance e is represented by a vector $(\mathbf{x}, c)$, where c is the class label. In general, we use a "hat" to indicate parameter estimates.

A Bayesian network encodes a joint probability distribution $P(\mathbf{X}, C)$ by a set of local distributions P for each variable. By forcing the class variable C to be the parent of each variable X_i, we can compute the posterior probability $P(C|\mathbf{X})$ as follows.

$$P(C|\mathbf{X}) = \alpha P(C) \prod_{i=1}^{n} P(X_i|\mathbf{U}_i), \tag{1}$$

where α is a normalization factor, and $\mathbf{U}_i$ denotes the set of parents of variable X_i. Note that the class variable C is always one parent of X_i. In naive Bayes, $\mathbf{U}_i$ only contains the class variable C. $P(C)$ is called the prior probability and $P(X_i|\mathbf{U}_i)$ is called the local probability distribution of X_i.

The local distribution $P(X_i|\mathbf{U}_i)$ is usually represented by a conditional probability table (CPT), which enumerates all the conditional probabilities for each assignment of values to X_i and its parents $\mathbf{U}_i$. Each conditional probability $P(x_{ij}|\mathbf{u}_{ik})$ in a CPT is often estimated using the corresponding frequencies obtained from the training data as follows.

$$\hat{P}(x_{ij}|\mathbf{u}_{ik}) = \frac{n_{ijk}}{n_{ik}}, \tag{2}$$

where n_{ijk} denotes the number of training instances in which variable X_i takes on the value x_{ij} and its parents $\mathbf{U}_i$ take on the values $\mathbf{u}_{ik}$. n_{ik} is equal to the sum of n_{ijk} over all j. The prior probability $P(C)$ is also estimated in the same way.

For the convenience in implementation, an entry θ_{ijk} in a CPT is the frequency n_{ijk}, instead of $P(x_{ij}|\mathbf{u}_{ik})$, which can be easily converted to $P(x_i|\mathbf{u}_i)$. To compute the frequencies from a given training data set, we go through each training instance, and increase the corresponding entries θ_{ijk} in CPTs by 1. By scanning the training data set once, we can obtain all the required frequencies and then compute the corresponding conditional probabilities. This parameter learning method is called *Frequency Estimate* (FE).

It is well-known that FE is a generative learning approach, because it maximizes likelihood (Friedman et al., 1997). In classification, however, the parameter setting that maximizes generalization accuracy is desired. Theoretically, if the structure of a Bayesian network is correct, the parameters determined by FE also maximize generalization accuracy. In practice, however, this assumption is rarely true. Therefore, the parameter learning method that directly maximizes generalization accuracy is more desirable in classification.

4. Discriminative Frequency Estimate

We now introduce *Discriminative Frequency Estimate* (DFE), a discriminative parameter learning algorithm for Bayesian network classifiers.

Note that, when counting a training instance in FE, we simply increase the corresponding frequencies by 1. Consequently, we do not directly take the effect on classification into account in computing frequencies. In fact, at any step in this process, we actually have a classifier on hand: the classifier whose local probabilities are computed by Equation 2 using the current entries (frequencies) in CPTs.

Thus, when we count an instance, we can apply the current classifier to it, and then update the corresponding entries based on how well (bad) the current classifier predicts on the instance. Intuitively, if the instance can be classified perfectly, there is no need to change any entries. In general, given an instance e, we can compute the difference between the true probability $P(c|e)$ and the predicted probability $\hat{P}(c|e)$ generated by the current parameters, where c is the true class of e, and then update the corresponding entries based on the difference. Furthermore, the FE process can be generalized such that we can count each instance more than once (as many as needed) until an convergence occurs. This is the basic idea of DFE.

More precisely, the DFE parameter learning algorithm iterates through the training instances. For each instance e, DFE firstly computes the predicted probability $\hat{P}(c|e)$, and then updates the frequencies in corresponding CPTs using the difference between the true $P(c|e)$ and the predicted $\hat{P}(c|e)$. The detail of the algorithm is depicted as follows. Here M is a pre-defined maximum number of steps. $L(e)$ is the prediction loss for training instance e based on the current parameters Θ^t, defined as follows.

$$L(e) = P(c|e) - \hat{P}(c|e). \qquad (3)$$

In general, $P(c|e)$ are difficult to know in classification task, because the information we have for c is only the class label. Thus, we assume that $P(c|e) = 1$ when e is in class c in our implementations. Note this assumption may not be held if data can not be separated completely, and thus may introduce bias to our probability estimation.

Algorithm 1 Discriminative Frequency Estimate

```
1. Initialize each CPT entry $\theta_{ijk}$ to 0
2. For $t$ from 1 to $M$ Do
     • Randomly draw a training instance $e$ from the training data set D.
     • Compute the posterior probability $\hat{P}(c|e)$ using the current parameters $\Theta^t$ and Equation 2.
     • Compute the loss $L(e)$ using Equation 3.
     • For each corresponding frequency $\theta_{ijk}$ in CPTs
         – Let $\theta_{ijk}^{t+1}=\theta_{ijk}^{t}+L(e)$.
```

Note that, in the beginning, each CPT entry θ_{ijk} is 0, and thus the predicted $\hat{P}(c|e)$ is $\frac{1}{|C|}$ after the probability normalization. In each step, if the current parameters Θ^t cannot accurately predict $P(c|e)$ for an instance e, the corresponding entries θ_{ijk} are increased significantly. If the current parameter Θ^t can perfectly predict $P(c|e)$, there will be no change on any entry.

The following summarizes our understanding for DFE:

1. The generative element is Equation 2. If we set the additive updates $L(e)$ in Equation 3 as a constant, DFE will be a maximum likelihood estimator, which is exactly the same as in the traditional naive Bayes. Thus, the parameters learned by DFE are influenced by the likelihood information $P(x_{ij}|\mathbf{u}_{ik})$ through Equation 2.

2. The discriminative element is Equation 3. If we use each entry θ_{ijk} in CPTs as parameters rather than generating the parameters using Equation 2, DFE will be a typical perceptron algorithm in the sense of error-driven learning. Thus, the parameters learned by DFE are also influenced by the prediction error through Equation 3.

3. DFE is different from a perceptron algorithm because of Equation 2. As we explained above, if we set the additive updates $L(e)$ in Equation 3 as a constant, there is no difference between DFE and a traditional naive Bayes. However, if we set the additive updates in a standard perceptron algorithm as a constant, the perceptron algorithm will not learn a traditional naive Bayes.

In summary, DFE learns parameters by considering the likelihood information $P(x_{ij}|\mathbf{u}_{ik})$ and the prediction error $P(c|e) - \hat{P}(c|e)$, and thus can be considered as a combination of generative and discriminative learning. Moreover, the likelihood information $P(x_{ij}|\mathbf{u}_{ik})$ seems to be more important than $P(c|e) - \hat{P}(c|e)$. For example, a DFE algorithm without Equation 2 performs significantly worse than naive Bayes, while a DFE algorithm without Equation 3 can still learn a traditional naive Bayes.

5. An Example

Before presenting our experiments, it could be helpful to get some intuitive feeling on DFE through a simple example.

A1	A2	A3	C
1	1	1	-
1	1	1	-
0	0	0	+
0	0	0	-
0	0	0	-

Figure 1. A data set with duplicate variables

Figure 1 shows a learning problem consisting of 5 instances and 3 variables. The variables A_2 and A_3 are the two duplicates of A_1, and thus all variables are perfectly dependent. For an instance $e = \{A_1 = 0, A_2 = 0, A_3 = 0\}$, the true posterior probability ratio is:

$$\frac{p(C = +|A_1 = 0, A_2 = 0, A_3 = 0)}{p(C = -|A_1 = 0, A_2 = 0, A_3 = 0)} = \frac{1}{2} \tag{4}$$

However, naive Bayes, which does not consider the dependencies between variables, gives the estimated posterior probability ratio:

$$\frac{\hat{p}(C = +)}{\hat{p}(C = -)}\left(\frac{\hat{p}(A_1 = 0|+)}{\hat{p}(A_1 = 0|-)}\right)^3 = \frac{2}{1} \tag{5}$$

Thus, naive Bayes misclassifies e. Moreover, the estimated posterior probability $\hat{p}(C = +|A_1 = 0, A_2 = 0, A_3 = 0)$ from naive Bayes is 0.66, while the true probability $p(C = +|A_1 = 0, A_2 = 0, A_3 = 0) = 0.33$. This mismatch is due to the two duplicates A_2 and A_3. Since $\frac{p(A_i=0|C=+)}{p(A_i=0|C=-)} = 2$, the duplication of A_1 results in overestimating the probability that e belongs to the positive class.

For DFE, the story is different. Figure 2 shows how the estimated probability $\hat{p}(C = +|A_1 = 0, A_2 = 0, A_3 = 0)$ in naive Bayes changes with FE and DFE respectively, as the number of instances used increases. Both algorithms take an instance in the order in Figure 1 at each step, and update the corresponding frequencies. With the increased number of instances used, the estimated probability $\hat{p}(C = +|A_1 = 0, A_2 = 0, A_3 = 0)$ from DFE converges to 0.4 approximately, which is close to the true probability and leads to a correct classification. However, FE converges to 0.66, even using the training instances more than once.

From this example, we can see that computing the frequencies in a discriminative way tends to yield more accurate probability estimation and give more accurate classification consequently. Also, both DFE and FE tend to converge with the increased training effort.

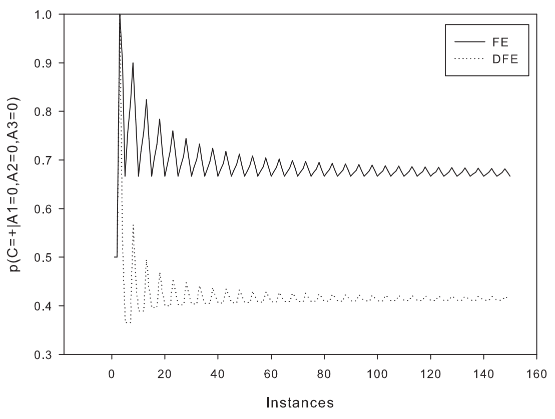

Figure 2. The y-axis is the predicted probability. The x-axis is the t_{th} instance fed into the algorithms.

6. Experiments

6.1. Experimental Setup

We conduct our experiments under the framework of WEKA (Witten & Frank, 2000). All experiments are performed on a Pentium 4 with 2.8GHZ CPU and 1G RAM. In our experiments, we use the 33 UCI data sets, selected by WEKA, which represent a wide range of domains and data characteristics. The smallest training data set "labor" has 51 training instances, and the largest data set "mushroom" has 7311 training instances. Numeric variables are discretized using the unsupervised ten-bin discretization implemented in WEKA. Missing values are replaced with the mean values from the training data. The multi-class data sets are transformed into binary ones by taking the two largest classes. The performance of an algorithm on each data set is observed via 10 runs of 10-fold stratified cross validation.

Table 1. Experimental results on accuracy

Data set	NB+DFE	NB+FE	NB+ELR	NB+Ada	HGC+FE	HGC+DFE
Labor	92.73±12.17	96.27± 7.87	95.53± 9.00	86.53±13.95	89.80±10.80	86.93±12.12
Zoo	100.00± 0.00	100.00± 0.00	100.00± 0.00	100.00± 0.00	100.00± 0.00	100.00± 0.00
Iris	100.00± 0.00	100.00± 0.00	96.20±11.05	100.00± 0.00	100.00± 0.00	100.00± 0.00
Primary-tumor	84.12± 9.17	84.12± 9.48	83.32± 9.99	80.82± 8.77	82.94± 9.07	82.23±10.32
Autos	88.94± 9.81	77.24±12.03 ●	90.27± 9.08	88.76± 8.70	83.97±10.75	84.49±11.85
Audiology	100.00± 0.00	99.82± 1.29	97.31± 5.22	99.82± 1.29	99.82± 1.29	100.00± 0.00
Glass	80.45± 9.91	76.37±10.59	81.44±10.04	75.03± 9.12	72.12±11.89 ●	71.55±12.32 ●
Vowel	95.89± 4.87	83.56± 8.76 ●	92.33± 5.71	94.44± 4.63	97.44± 3.22	97.44± 3.41
Soybean	98.58± 3.30	95.52± 4.74	97.29± 4.13	97.38± 3.18	97.49± 3.85	98.05± 3.27
Hepatitis	84.79± 9.11	84.13±10.34	83.49±10.41	81.42± 9.33	83.01± 9.04	83.71± 9.01
Sonar	76.85± 9.30	76.02±10.67	77.36± 9.49	75.23± 9.09	69.16±10.44	68.89±10.49
Lymphography	86.33± 8.95	86.21± 8.12	85.08± 8.84	83.34± 9.56	84.40± 9.10	84.23± 8.49
Heart-statlog	82.89± 5.69	83.70± 5.60	82.96± 5.80	76.44± 7.59 ●	83.04± 5.55	82.00± 4.96
Cleveland	83.04± 7.49	83.57± 5.99	82.50± 7.11	79.08± 7.94 ●	82.32± 7.46	80.99± 7.43
Breast-cancer	70.36± 8.05	72.87± 7.48	71.34± 8.04	69.73± 7.71	74.26± 5.45	73.49± 5.98
Ionosphere	90.54± 5.32	90.83± 3.99	91.11± 4.82	89.13± 6.14	93.28± 4.53	91.51± 5.29
Horse-colic	82.99± 6.03	78.70± 6.27 ●	80.59± 6.71	77.60± 6.30 ●	83.70± 5.30	82.01± 6.86
Vehicle	93.74± 3.40	82.82± 6.80 ●	92.96± 3.52	93.84± 3.31	95.68± 3.11	96.78± 2.87 ○
Vote	94.80± 2.86	90.29± 4.07 ●	95.72± 2.87	94.39± 3.12	94.84± 3.15	95.44± 3.15
Balance	99.48± 0.94	99.24± 1.17	99.83± 0.52	99.27± 1.17	99.27± 1.22	99.69± 0.76
Wisconsin	96.45± 2.05	97.31± 1.70	96.22± 2.10	95.11± 2.40	96.91± 1.51	96.71± 2.11
Segment	100.00± 0.00	100.00± 0.00	100.00± 0.00	100.00± 0.00	100.00± 0.00	100.00± 0.00
Credit-rating	85.51± 3.65	84.75± 3.68	85.25± 4.16	82.96± 4.08 ●	85.51± 4.27	86.03± 3.80
Diabetes	75.78± 4.67	75.57± 4.76	76.15± 4.36	74.90± 4.75	75.94± 5.14	75.60± 4.68
Anneal	99.72± 0.74	97.67± 1.98 ●	99.87± 0.53	99.87± 0.59	98.95± 1.20	99.92± 0.30
Credit-g	75.98± 4.01	75.90± 3.97	75.64± 3.73	74.22± 4.43	72.88± 3.88 ●	72.92± 3.53 ●
Letter	99.52± 0.49	96.49± 1.42 ●	98.95± 0.74 ●	98.94± 0.83 ●	98.42± 1.01 ●	99.46± 0.52
Splice	97.62± 0.98	97.61± 0.98	97.64± 0.89	95.70± 1.41 ●	98.05± 0.91	98.01± 0.85
Kr-vs-kp	94.70± 1.37	87.80± 1.89 ●	95.68± 1.21 ○	95.19± 1.19	92.40± 1.61 ●	95.54± 1.37 ○
Waveform	91.05± 1.52	87.52± 1.46 ●	90.71± 1.16	89.19± 1.63 ●	89.66± 1.40 ●	89.72± 1.43 ●
Hypothyroid	95.95± 0.46	95.49± 0.49 ●	95.83± 0.49	95.55± 0.66 ●	95.72± 0.52	95.88± 0.51
Sick	97.49± 0.67	96.75± 0.97 ●	97.47± 0.74	96.79± 0.83 ●	97.82± 0.75	97.91± 0.70
Mushroom	99.96± 0.06	95.53± 0.63 ●	100.00± 0.00	100.00± 0.00	99.96± 0.06	100.00± 0.00

● worse, and ○ better, comparing with NB-DFE.

Two Bayesian network classifiers, naive Bayes (NB) and HGC (Heckerman et al., 1995), are used to compare the performance of different parameter learning methods. HGC is a hill-climbing structure search algorithm. In our experiments with HGC, we limit the number of parents of each node to 2.

In general, we use NB+X and HGC+X to indicate that NB and HGC with a specific parameter learning method X respectively: X is one of FE, DFE, ELR and Ada (Freund & Schapire, 1996). Note that, for HGC+DEF, we use HGC to learn the structure first, and then apply DFE to learn parameters. We do not use DFE in the structure learning of HGC. The following summarizes the parameter learning algorithms used in our experiments.

FE: the *generative parameter learning* method. Note the term "one iteration" in this paper indicates that we count all training instances exactly once.

DFE: the *discriminative parameter learning* method, depicted in Section 4. In our implementation, we simply go through the whole training data four times (iterations), instead of randomly choosing instances.

ELR: the *gradient descent based discriminative parameter learning* method, proposed in (Greiner & Zhou, 2002).

Ada: *Adaboost M1* is used as an ensemble method that combines the outputs of base classifiers to produce a better prediction (Freund & Schapire, 1996). The number of classifiers is 20.

In our experiments, we use the implementation of ELR from the authors (Greiner & Zhou, 2002) and the implementation of HGC and Ada in WEKA, and implement DFE in WEKA.

Table 2. Summary of the experimental results on accuracy.

	NB+FE	NB+ELR	NB+Ada	HGC+FE	HGC+DFE
NB+DFE	12/21/0	1/31/1	9/24/0	5/28/0	3/28/2
NB+FE		0/22/11	9/19/5	0/22/11	1/22/10
NB+ELR			4/29/0	4/27/2	2/28/3
NB+Ada				2/26/5	0/28/5
HGC+FE					0/30/3

6.2. Accuracy and Training Time

Table 1 gives the detailed experimental results on accuracy. To better understand the effect of training data size on the algorithm performance, we sort the data sets by their sizes. Table 2 shows the results of the paired t-test with significance level 0.05, in which each entry $w/t/l$ means that the learner in the corresponding row wins in w data sets, ties in t data sets, and loses in l data sets, compared to the learning algorithm in the corresponding column. The following is the highlight of our observations.

1. The two discriminative parameter learning methods ELR and DFE have the similar performance

Table 3. Experimental results on training time

Data set	NB+DFE	NB+FE	NB+ELR	NB+Ada	HGC+FE	HGC+DFE
Labor	0.0009±0.00	0.0002±0.00 ●	55.0250± 48.89 ○	0.0066±0.00 ○	0.0367±0.00 ○	0.0416±0.00 ○
Zoo	0.0006±0.00	0.0001±0.00 ●	100.1444± 49.28 ○	0.0018±0.00 ○	0.0077±0.00 ○	0.0120±0.00 ○
Iris	0.0005±0.00	0.0001±0.00 ●	317.4304± 150.43 ○	0.0019±0.00 ○	0.0030±0.00 ○	0.0058±0.00 ○
Primary-tumor	0.0010±0.00	0.0002±0.00 ●	99.7059± 12.72 ○	0.0097±0.00 ○	0.0178±0.00 ○	0.0295±0.00 ○
Autos	0.0017±0.00	0.0002±0.00 ●	202.3540± 42.84 ○	0.0213±0.02 ○	0.2843±0.03 ○	0.2956±0.04 ○
Audiology	0.0019±0.00	0.0003±0.00 ●	311.3375± 55.36 ○	0.0023±0.00	0.5920±0.05 ○	0.6141±0.05 ○
Glass	0.0008±0.00	0.0002±0.00 ●	205.4710± 14.92 ○	0.0117±0.00 ○	0.0363±0.00 ○	0.0464±0.02 ○
Vowel	0.0013±0.00	0.0003±0.00 ●	399.6574± 263.88 ○	0.0176±0.00 ○	0.0373±0.00 ○	0.0510±0.00 ○
Soybean	0.0018±0.00	0.0004±0.00 ●	507.1840± 55.93 ○	0.0230±0.01 ○	0.0464±0.00 ○	0.0705±0.02 ○
Hepatitis	0.0015±0.00	0.0002±0.00 ●	414.9584± 27.77 ○	0.0191±0.00 ○	0.0369±0.00 ○	0.0559±0.02 ○
Sonar	0.0066±0.00	0.0007±0.00 ●	932.8643± 106.34 ○	0.0669±0.02 ○	4.8039±0.21 ○	4.8389±0.20 ○
Lymphography	0.0037±0.02	0.0003±0.00	387.2173± 19.10 ○	0.0164±0.00 ○	0.0335±0.00 ○	0.0480±0.00 ○
Heart-statlog	0.0019±0.00	0.0003±0.00 ●	579.2737± 74.95 ○	0.0252±0.02 ○	0.0494±0.02 ○	0.0674±0.02 ○
Cleveland	0.0020±0.00	0.0028±0.02	681.2536± 109.79 ○	0.0209±0.01 ○	0.0239±0.00 ○	0.0451±0.00 ○
Breast-cancer	0.0015±0.00	0.0002±0.00 ●	541.8432± 56.39 ○	0.0126±0.00 ○	0.0161±0.02 ○	0.0288±0.00 ○
Ionosphere	0.0054±0.00	0.0007±0.00 ●	2261.0212± 780.54 ○	0.0629±0.02 ○	0.3492±0.04 ○	0.4219±0.04 ○
Horse-colic	0.0044±0.00	0.0005±0.00 ●	1506.9836± 146.88 ○	0.0430±0.01 ○	0.0987±0.02 ○	0.1457±0.02 ○
Vehicle	0.0039±0.00	0.0005±0.00 ●	2125.4934± 137.27 ○	0.0480±0.00 ○	0.1531±0.02 ○	0.2009±0.03 ○
Vote	0.0034±0.00	0.0005±0.00 ●	1779.7511± 251.58 ○	0.0334±0.02 ○	0.0229±0.02 ○	0.0632±0.02 ○
Balance	0.0017±0.00	0.0005±0.00 ●	2710.6686±1280.37 ○	0.0243±0.01 ○	0.0038±0.00 ○	0.0189±0.00 ○
Wisconsin	0.0034±0.00	0.0005±0.00 ●	1376.4606± 146.91 ○	0.0559±0.02 ○	0.0243±0.00 ○	0.0624±0.02 ○
Segment	0.0057±0.00	0.0008±0.00 ●	3973.2459± 659.38 ○	0.0039±0.00 ●	0.1233±0.02 ○	0.1952±0.03 ○
Credit-rating	0.0076±0.02	0.0006±0.00	1316.8793± 68.50 ○	0.0514±0.02 ○	0.0648±0.02 ○	0.1252±0.02 ○
Diabetes	0.0034±0.00	0.0005±0.00 ●	1118.3888± 41.75 ○	0.0344±0.01 ○	0.0299±0.00 ○	0.0676±0.02 ○
Anneal	0.0097±0.00	0.0011±0.00 ●	4947.6380±1573.55 ○	0.1098±0.03 ○	0.1797±0.03 ○	0.3056±0.03 ○
Credit-g	0.0103±0.00	0.0012±0.00 ●	2440.2377± 357.70 ○	0.0745±0.03 ○	0.1473±0.03 ○	0.2611±0.03 ○
Letter	0.0223±0.00	0.0024±0.00 ●	262.4565± 142.62 ○	0.3817±0.15 ○	0.2089±0.06 ○	0.5355±0.20 ○
Splice	0.1322±0.04	0.0143±0.02 ●	2398.4974± 835.69 ○	1.3441±0.44 ○	8.1985±2.28 ○	9.8085±2.38 ○
Kr-vs-kp	0.1533±0.08	0.0232±0.06 ●	1648.1174± 856.53 ○	1.2348±0.16 ○	1.0847±0.17 ○	1.9889±0.11 ○
Waveform	0.1829±0.05	0.0171±0.00 ●	2743.9441± 295.50 ○	1.5109±0.16 ○	2.3946±0.26 ○	3.4949±0.28 ○
Hypothyroid	0.1091±0.04	0.0121±0.01 ●	1035.1162± 543.62 ○	1.2212±0.63 ○	1.1376±0.34 ○	3.3269±1.24 ○
Sick	0.1246±0.08	0.0095±0.00 ●	2662.4956± 379.71 ○	0.6825±0.27 ○	1.6360±0.64 ○	3.4699±0.99 ○
Mushroom	0.2102±0.15	0.0205±0.02 ●	11243.5967±3074.40 ○	2.6704±0.95 ○	2.2242±0.86 ○	4.5443±1.32 ○

○ slower, and ● faster comparing with NB-DFE The training time unit is second

in terms of accuracy. NB+DFE performs better than NB+ELR in 1 data set and loses in 1 data set.

2. For naive Bayes, the discriminative parameter learning methods significantly improve the performance of the generative parameter learning method FE. NB+ELR and NB+DFE outperform NB+FE in 11 and 12 data sets without a loss respectively. In our experiments, NB+Ada loses to NB in 9 data sets and wins in 5 data sets. This means that using boosting as a discriminative parameter learning method is not effective according to our experiments.

3. NB+DFE outperforms HGC+FE in 5 data sets without a loss. Note that there is no structure learning in NB+DFE at all. Thus, we could expect that discriminative parameter learning can significantly reduce the effort for structure learning.

4. DFE improves the general Bayesian network learning algorithm HGC. HGC+DFE outperforms HGC+FE in 3 data sets without a loss. This improvement is not as significant as in naive Bayes. However, it is consistent with previous research results: while the structure of a Bayesian network is closer to the "true" one, discriminative parameter learning is less helpful (Greiner & Zhou, 2002; Grossman & Domingos, 2004).

5. HGC+FE outperforms NB+FE in 11 data sets without a loss. This results show that many data sets in our experiments contains strong dependencies. The structure learning in HGC relaxes the independence assumption in naive Bayes, and thus improves the performance significantly.

We have also observed the training time for each algorithm. Table 3 shows the average training time of each algorithm from 10 runs of 10-fold stratified cross validation. From Table 3, we can see that DFE is approximately 250,000 times faster than ELR. Recall that their performance in classification accuracy is similar. Certainly, FE is still the most efficient algorithm: 7 times faster than DFE, 70 faster than NB+Ada, and 1,800,000 times faster than ELR approximately.

6.3. Convergence, Overfitting and Learning Curves

In our experiments, we have investigated the convergence of the DFE algorithm. We have observed the relation between the number of iterations and the accuracy of NB+DFE on the 8 largest data sets, shown in Figure 3. Again, an iteration means counting all instances once. Each point in the curves corresponds to the number of iterations that a parameter learning method performs over the training data and the average accuracy from 10-fold cross validation.

Figure 3 shows that NB+DFE converges quickly. We

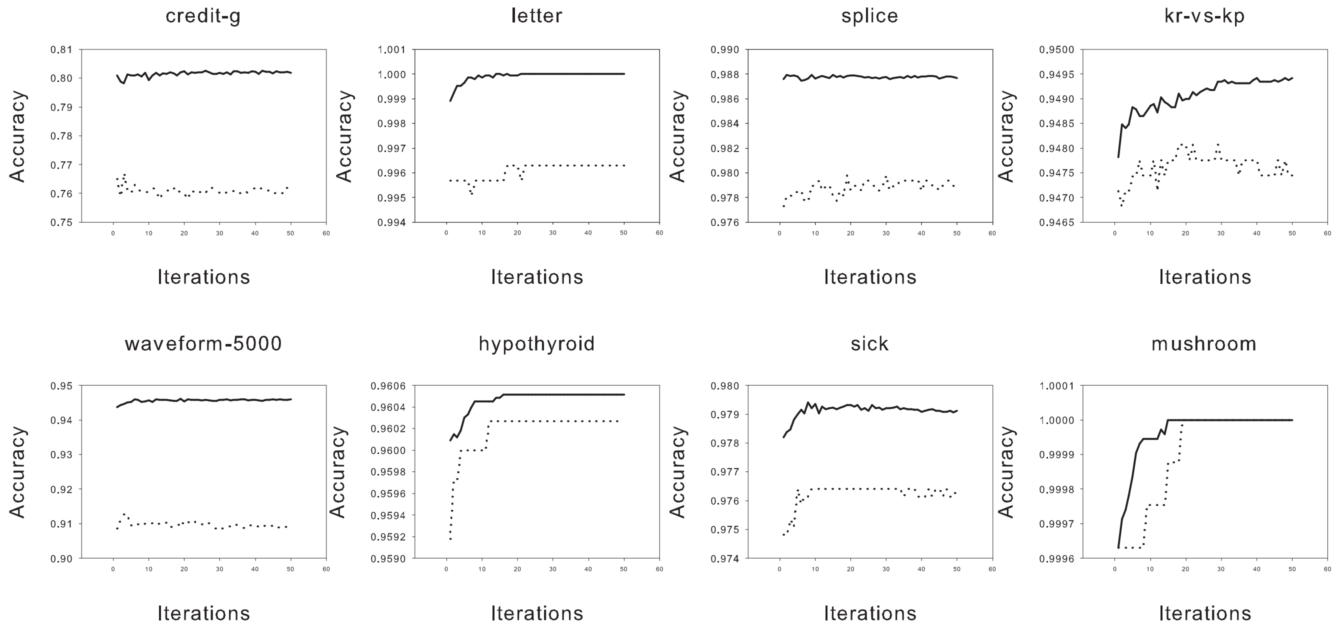

Figure 3. Relation between accuracies and the number of iterations over training and testing data. Solid lines represent training accuracy, and dotted lines represent testing accuracy.

can see that NB+DFE approaches its highest accuracy just after one iteration. As the number of iterations increases after that, there is no significant difference. For example, in all the 8 data sets, the differences between NB+DFE with one iteration and with more iterations are only around 0.005. In our experiments, in fact, we have tried different iteration numbers (1 to 2048) for DFE, and the accuracies of NB+DFE and HGC+DFE do not significantly change.

In the 33 data sets, there is only one data set "Vowel", in which NB+DFE needs more than one iteration to reach the asymptotic accuracy. NB+DFE achieves 90.00% after one iteration, and approaches 95.89% after 4 iterations. The "Vowel" data set has been observed to contain strong variable dependencies (Su & Zhang, 2005), and is small (contains only 180 training instances). However, when the sample size is not small, such as in "Kr-vs-kp" and "Mushroom", one iteration is still enough for DFE to reach its asymptotic accuracy, even though there are strong dependencies in these data sets.

From Figure 3, we can also observe that NB+DFE does not suffer from overfitting. With the increased iterations, the accuracies on test data, shown by the dotted lines, remain the same. That means, once NB+DFE reaches its asymptotic accuracy, the more learning effort does not influence the model. Consequently, no stopping criterion is required for DFE. In contrast, the discriminative learning algorithm ELR requires a stopping criterion to prevent overfitting. Greiner and Zhou (2002) showed that the accuracy of ELR may decrease with an increased training effort.

We have also studied the learning curves of NB+DFE. Ng and Jordan (2001) showed that discriminative learning may have disadvantage comparing to generative learning when sample size is small. Thus, we are interested in how our discriminative parameter learning algorithm DFE performs in this scenario.

Figure 4 shows the learning curves for NB+FE, NB+ELR, and NB+DFE on the same 8 UCI data sets. Since we are interested in the performance in a small sample size, we only observe the performance of each algorithm using up to 50 instances. The accuracy in the learning curves is the average accuracy, obtained on the data that is not used for training with a total of 30 runs. The learning curves show how the accuracy changes as more labeled data are used.

From Figure 4, we can see that NB+FE dominates NB+DFE and NB+ELR only on data sets "Credit-g" and "Hypothyroid" in terms of accuracy. On data sets "Kr-vs-kp" and "Mushroom", however, both discriminative learning algorithms NB+ELR and NB+DFE outperform NB+FE. On all other data sets, the results are mixed. It means that generative learning has actually no obvious advantage over discriminative learning even when the size of training data is small. In fact, our observations agree with the analysis in (Greiner & Zhou, 2002).

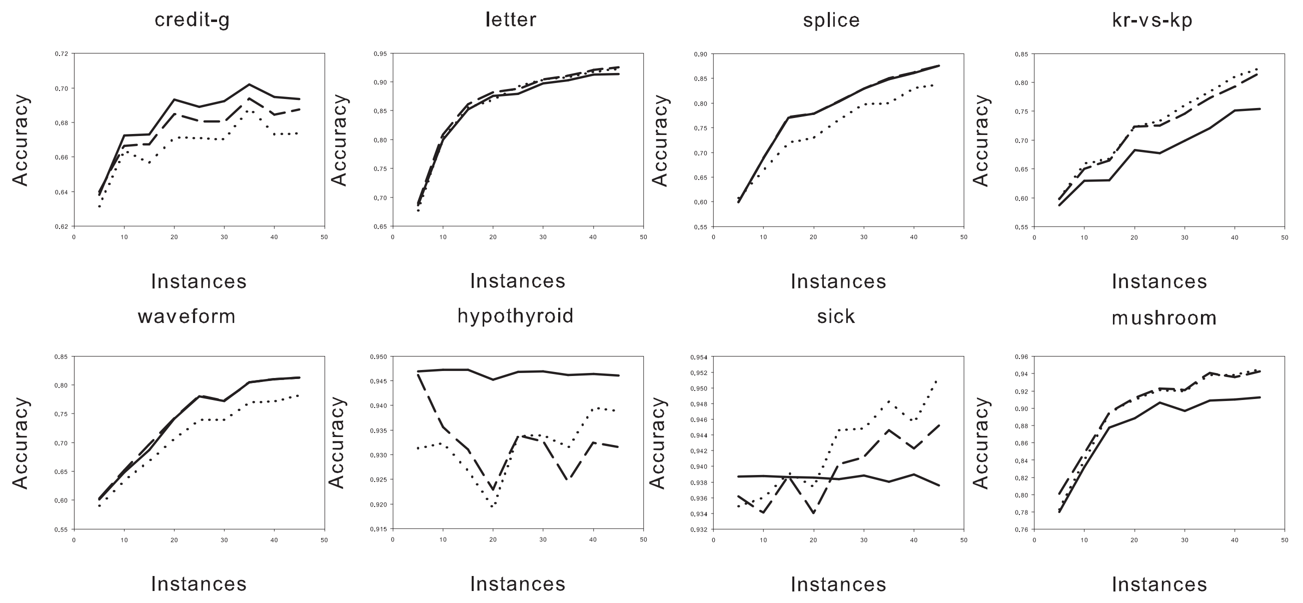

Figure 4. Relation between accuracies and training data sizes. Solid, dotted, and dashed lines correspond to NB+FE, NB+DFE, and NB+ELR respectively.

7. Conclusion

In this paper, we propose a novel discriminative parameter learning method for Bayesian network classifiers. DFE can be viewed as a discriminative version of frequency estimate. Our experiments show that the DFE algorithm combines the advantages of generative and discriminative learning: it is computationally efficient, converges quickly, does not suffer from the overfitting problem, and performs competitively with the state-of-the-art discriminative parameter learning algorithm ELR in accuracy.

This paper mainly studies the empirical side of DFE. Its theoretical nature remains unknown. Moreover, because of the efficiency of DFE, we would expect that DFE could be applied in general structure learning, leading to more accurate Bayesian network classifiers. In our future work, we will study DFE from theoretical perspective and embed DFE into the structure search process of HGC and other structure learning algorithms.

References

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proceedings of the Thirteenth International Conference on Machine Learning* (pp. 148 – 156).

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*, 131–163.

Greiner, R., & Zhou, W. (2002). Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers. *AAAI/IAAI* (pp. 167–173).

Grossman, D., & Domingos, P. (2004). Learning bayesian network classifiers by maximizing conditional likelihood. *ICML '04: Proceedings of the twenty-first international conference on Machine learning* (p. 46). New York, NY, USA: ACM Press.

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

Ng, A. Y., & Jordan, M. I. (2001). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *NIPS* (pp. 841–848).

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems:networks of plausible inference.* Morgan Kauhmann.

Su, J., & Zhang, H. (2005). Representing conditional independence using decision trees. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 874–879. AAAI Press.

Witten, I. H., & Frank, E. (2000). *Data mining – practical machine learning tools and techniques with Java implementation.* Morgan Kaufmann.