Bayesian Classification for Data From the Same Unknown Class

Hung-Ju Huang and Chun-Nan Hsu, *Member, IEEE*

***Abstract*—In this paper, we address the problem of how to classify a set of query vectors that belong to the same unknown class. Sets of data known to be sampled from the same class are naturally available in many application domains, such as speaker recognition. We refer to these sets as *homologous sets*. We show how to take advantage of homologous sets in classification to obtain improved accuracy over classifying each query vector individually. Our method, called *homologous naive Bayes (HNB)*, is based on the naive Bayes classifier, a simple algorithm shown to be effective in many application domains. HNB uses a modified classification procedure that classifies multiple instances as a single unit. Compared with a voting method and several other variants of naive Bayes classification, HNB significantly outperforms these methods in a variety of test data sets, even when the number of query vectors in the homologous sets is small. We also report a successful application of HNB to speaker recognition. Experimental results show that HNB can achieve classification accuracy comparable to the Gaussian mixture model (GMM), the most widely used speaker recognition approach, while using less time for both training and classification.**

***Index Terms*—Classification, machine learning, naive Bayes classifier, speaker recognition.**

I. Introduction

THE PROBLEM of classification is actively researched in pattern recognition and machine learning. Research on classification centers on developing classification systems that correctly recognize unknown patterns. In classical pattern recognition [1], the input to a classifier is a single query vector and the output is a class label for that query vector. However, suppose we know that a set of query vectors belong to the same class. For example, a botany student discovers a plant she has never seen before. In order to identify the plant's species, she collects several leaves to provide input to a classifier. Obviously, the output class labels for these leaves should all be the same. We refer to a set of query vectors sampled from the same class as a *homologous set*. Such samples are readily available in many other applications, such as speaker recognition [2]. The combined information from multiple query vectors in a homologous set might be used to aid the classification process. However, standard classification algorithms are not designed to take advantage of this knowledge.

Due to the robust performance and simple implementation, the naive Bayes classifier has become a popular classification tool in recent years. Many applications prefer the naive Bayes classifier because of its simplicity. In spite of its simplicity, the naive Bayes classifier achieves comparable performance with popular classifiers such as C4.5 [3],and constantly outperforms competing algorithms, on average, in experiments reported in the literature [4]. Remarkably, in KDD-CUP-97, two of the top three contestants are based on the naive Bayes classifier [5]. Also, Domingos and Pazzani [6] reported an experiment that compared the naive Bayes classifier with several classical learning algorithms on a large ensemble of data sets. Their results also show that the naive Bayes classifier is a good classification tool.

Previous work has focused on improving the accuracy of naive Bayesian classifiers with a single query vector. No past study has been devoted to the problem of classifying homologous sets with the naive Bayes classifiers. In this paper, we present a method called *homologous naive Bayes (HNB)*, which allows for the efficient classification of homologous sets by the naive Bayes classifier. We empirically compared this method with voting and several other extensions of the naive Bayes classifier, which we will describe later. Experimental results show that HNB outperforms all other methods. Further analysis shows that, to improve the classification accuracy, the other extension methods require large homologous sets, while HNB can significantly improve the result accuracy even when there is only one pair of query vectors in each homologous set.

Our application of the HNB method to speaker recognition proved to be successful. In this type of application, we usually have prior information that large sets of query vectors come from the same unknown speaker.

The remainder of this paper is organized as follows: Section II reviews the naive Bayes classifier. Section III presents several approaches to classifying homologous sets. Section IV empirically compares the performance of the different methods. Section V reports an application of HNB to speaker recognition. Finally, Section VI contains the summary of conclusions.

II. Naive Bayes Classifiers

The naive Bayes classifier is based on the simplifying assumption that the feature values are conditionally independent given the class label [7]. Fig. 1 gives a graphical depiction. A naive Bayes classifier classifies a query vector $\mathbf{x}$ of predictive

Manuscript received March 14, 2001; revised August 2, 2001, October 9, 2001, and November 6, 2001. This work was supported in part by the National Science Council of Taiwan, R.O.C., under Grant NSC 89-2213-E-001-031. The speech data set was courtesy of the Speech Processing Laboratory, Department of Communication Engineering, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C. This paper was recommended by Associate Editor D. Cook.

H.-J. Huang is with the Department of Computer and Information Science, National Chiao-Tung University, Hsinchu 300, Taiwan, R.O.C.

C.-N. Hsu is with the Institute of Information Science, Academia Sinica, Nankang 115, Taipei City, Taiwan, R.O.C. (e-mail: chunnan@iis.sinica.edu.tw; http://chunnan.iis.sinica.edu.tw).

Publisher Item Identifier S 1083-4419(02)00696-9.

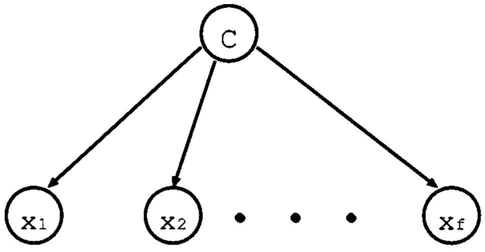

Fig. 1. Naive Bayes classifier, where the predictive features ($x_1, x_2, \ldots, x_f$) are conditionally independent given the class attribute (c).

features by selecting class c that maximizes the posterior probability

$$p(c|\mathbf{x}) \propto p(c) \prod_{x \in \mathbf{x}} p(x|c) \tag{1}$$

where x is a predictive feature (variable) in $\mathbf{x}$ and $p(x|c)$ is the *class-conditional density* of x given class c. Let $\boldsymbol{\theta}$ denote the vector whose elements are the parameters of the density of $p(x|c)$. In a Bayesian learning framework, we assume that $\boldsymbol{\theta}$ can be learned from a training data set. This estimation is at the heart of training in the naive Bayes classifier.

The naive Bayes classifier can handle discrete variables and continuous variables when assuming that their priors are Dirichelet distribution and normal distribution, respectively [8]. In the following section, we describe the learning procedures of a naive Bayes classifier with different types of variables.

A. Naive Bayes Classifier With Discrete Variables

Suppose x is a discrete variable with k possible values. In principle, the class label c of the data vector $\mathbf{x}$ dictates the probability of the value of x. Thus, the appropriate probability distribution function is a multinomial distribution and its parameters are a set of probabilities $\{\theta_1, \theta_2, \ldots \theta_k\}$, such that for each possible value X_j, $p(x = X_j|c) = \theta_j$ and $\sum_{j=1}^{k} \theta_j = 1$. Now, let $\boldsymbol{\theta} \equiv (\theta_1, \theta_2, \ldots \theta_k)$, a Dirichlet distribution [9] with parameters $\alpha_1, \ldots, \alpha_k$ as the prior for $\boldsymbol{\theta}$. Given a training data set, we can update $p(x = X_j|c)$ using its expected value

$$\hat{p}(x = X_j|c) = \frac{\alpha_j + y_{cj}}{\alpha + n_c} \tag{2}$$

where n_c is the number of the training examples belonging to class c, y_{cj} is the number of class c examples whose $x = X_j$, and $\alpha = \alpha_1 + \cdots + \alpha_k$. Since a Dirichlet distribution is conjugate to multinomial sampling, after the training, the posterior distribution of $\boldsymbol{\theta}$ is still a Dirichlet, but with the updated parameters being $\alpha_j + y_{cj}$ for all j. This property allows us to incrementally train the naive Bayes classifier.

In practice, we usually choose the Jaynes prior [10] $\alpha_j = \alpha = 0$ for all j and have $\hat{p}(x|c) = y_{cj}/n_c$. However, when the training data set is too small, this often yields $\hat{p}(x|c) = 0$ and impedes the classification. To avoid this problem, another popular choice is $\alpha_j = 1$ for all j. This is known as *smoothing* or *Laplace's estimate* [11].

B. Naive Bayes Classifier With Continuous Variables

If x is a continuous variable, a conventional approach is to assume that $p(x|c) = N(x; \mu_c, \sigma_c^2)$, where $N(x; \mu_c, \sigma_c^2)$ is the probability distribution function of a normal distribution. In this case, training involves learning the parameters μ_c and σ_c from the training data [8]. This approach has been shown to be less effective than discretization when $p(x|c)$ is not normal, and discretization is often used. Generally, discretization involves partitioning the domain of x into k intervals as a pre-processing step. Then we can treat x as a discrete variable with k possible values and conduct the training and classification as described in Section II-A.

More precisely, let I_j be the jth discretized interval. Training and classifying in the naive Bayes classifier with discretization is to use $\hat{p}(x \in I_j|c)$ as an estimate of $\hat{p}(x|c)$ in (1) for each continuous variable. This is equivalent to assuming that after discretization, the class-conditional density of x has a Dirichlet prior. This assumption is called *Dirichlet discretization assumption*. This assumption holds for all well-known discretization methods, including ten-bin, entropy-based [12], among others. (see [13] for a comprehensive survey).

Since it has been shown to be less effective than discretization when the distribution of a continuous variable is not normal, we must select a discretization method for our experiments. In our previous work [14], we explained why well-known discretization methods, such as entropy-based, bin-$\log l$ and ten-bin, work well for naive Bayes classifiers with continuous variables, regardless of their complexities. In this paper, we used the simple method "ten-bin" for partitioning continuous variables in all of our experiments. This method merely divides the range of observed values for a variable into ten equal-size bins. However, other discretization methods can also be used.

III. Classifying a Homologous Set

We start by showing that a classifier must deliberately take advantage of the knowledge that all data have the same unknown class label; otherwise, the knowledge will not improve the expected accuracy, and this is generally the case regardless of the number of query vectors in the homologous set and the number of classes that we want for classifying the data. Consider a classifier which classifies one query vector into one of the n classes, with accuracy σ. Suppose this classifier classifies m query vectors individually. The expected value of the accuracy can be derived from the following:

$$E_1 = \sum_{i=0}^{m} \frac{m-i}{m} p(y_i) \tag{3}$$

$$= \sum_{i=0}^{m} \frac{m-i}{m} \binom{m}{m-i} \sigma^{m-i}(1-\sigma)^i \tag{4}$$

$$= \sum_{i=0}^{m-1} \frac{m-i}{m} \frac{m!}{(m-i)!i!} \sigma^{m-i}(1-\sigma)^i \tag{5}$$

$$= \sum_{i=0}^{m-1} \frac{(m-1)!}{(m-1-i)!i!} \sigma^{m-i}(1-\sigma)^i \tag{6}$$

$$= \sum_{i=0}^{m-1} \binom{m-1}{m-1-i} \sigma^{m-i}(1-\sigma)^i \tag{7}$$

$$= \sigma \sum_{i=0}^{m-1} \binom{m-1}{m-1-i} \sigma^{m-1-i}(1-\sigma)^i \tag{8}$$

$$= \sigma \sum_{i=0}^{z} \binom{z}{z-i} \sigma^{z-i}(1-\sigma)^i \tag{9}$$

$$= \sigma(\sigma + (1-\sigma))^m \tag{10}$$

$$= \sigma \tag{11}$$

where y_i denotes the event that there are i query vectors which are classified incorrectly, and the expected value is the weighted average of the probabilities of y_i. "$m-1$" is replaced by z from (8) and (9) and the binomial theorem [15] is used to derive (10) from (9).

On the other hand, suppose we know that the m vectors actually belong to the same class and take them as one object. We therefore classify this object using the same classifier. The expected value of the accuracy is

$$E_2 = \frac{m}{m} p(y_0) + \frac{0}{m} p(y_1) \tag{12}$$

$$= \frac{m}{m} \binom{1}{1} \sigma(1-\sigma)^0 + \frac{0}{m} \binom{1}{0} \sigma^0(1-\sigma) \tag{13}$$

$$= \sigma. \tag{14}$$

The expected values of the above two cases are the same. This implies that the prior information, indicating that query vectors come from the same class, does not automatically improve the accuracy.

A. Voting, Averaging, Maximum, and Their Variations

There are several intuitive extensions for the naive Bayes classifier to classify homologous sets. One method is voting, which uses the naive Bayes classifier to classify each member in the homologous set and selects the class label predicted most often.

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be a homologous set of query vectors with the same unknown class label c and each query vector $\mathbf{x}_t$ in $\mathbf{X}$ has r features $(x_{t1}, \ldots, x_{tr})$. We assume that $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are drawn independently,[1] and the symbol O denotes the prior information that all members in the homologous set have the same unknown class label. According to the Bayesian decision theory, we should classify this homologous set by selecting the class c that maximizes $p(c|\mathbf{X}, O)$, the probability of c given $\mathbf{X}$ and O. We can derive four "extension" methods to estimate $p(c|\mathbf{X}, O)$ for classifying the query vectors in a homologous set.

1) **Avg**

The "Avg" method estimates $p(c|\mathbf{x}_t)$ for each $\mathbf{x}_t \in \mathbf{X}$ and averages the results to obtain $p(c|\mathbf{X}, O)$ as follows:

$$p(c|\mathbf{X}, O) \propto \frac{\sum_{t=1}^{n} p(c) \prod_{l=1}^{r} p(x_{tl}|c)}{n}. \tag{15}$$

2) **Local Avg** (LAvg)

In this method, for each feature $\mathbf{x}_l$, we compute the average of the class-conditional probability $p(x_{tl}|c)$ of all members $\mathbf{x}_t \in \mathbf{X}$ and use the result as the class-conditional probability $p(\mathbf{x}_l|c)$ of the feature $\mathbf{x}_l$. Then we can obtain $p(c|\mathbf{X}, O)$ as follows:

$$p(c|\mathbf{X}, O) \propto p(c) \prod_{l=1}^{r} \frac{\sum_{t=1}^{n} p(x_{tl}|c)}{n}. \tag{16}$$

3) **Max**

The "Max" method estimates $p(c|\mathbf{x}_t)$ for each $\mathbf{x}_t \in \mathbf{X}$ and selects the maximum probability as $p(c|\mathbf{X}, O)$ as follows:

$$p(c|\mathbf{X}, O) \propto \max_{t=1}^{n} \left(p(c) \prod_{l=1}^{r} p(x_{tl}|c) \right). \tag{17}$$

4) **Local Max** (LMax)

In this method, the class-conditional probability $p(\mathbf{x}_l|c)$ for each feature $\mathbf{x}_l$ is estimated by selecting the maximum $p(x_{tl}|c)$ among all members $\mathbf{x}_t \in \mathbf{X}$. Then we can obtain $p(c|\mathbf{X}, O)$ as follows:

$$p(c|\mathbf{X}, O) \propto p(c) \prod_{l=1}^{r} \max_{t=1}^{n}(p(x_{tl}|c)). \tag{18}$$

The classification rule of the above methods is to pick the class c that maximizes $p(c|\mathbf{X}, O)$.

B. Homologous Naive Bayes (HNB)

The above four methods are based on the idea that we can combine the estimation of $p(c|\mathbf{x}_t)$ to obtain $p(c|\mathbf{X}, O)$, by averaging or by selecting the maximum values. In fact, we can derive a method purely from the Bayes rule and independence assumptions as follows:

$$p(c|\mathbf{X}, O) = p(c|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, O) \tag{19}$$

$$= \frac{p(c)p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, O|c)}{p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, O)} \tag{20}$$

$$\propto p(c)p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, O|c) \tag{21}$$

$$= p(c)p(\mathbf{x}_1|c)p(\mathbf{x}_2|c, \mathbf{x}_1)p(\mathbf{x}_3|c, \mathbf{x}_1, \mathbf{x}_2)\cdots p(\mathbf{x}_n|c, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n-2}, \mathbf{x}_{n-1}) p(O|c, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n-1}, \mathbf{x}_n) \tag{22}$$

$$= p(c)p(\mathbf{x}_1|c)p(\mathbf{x}_2|c)p(\mathbf{x}_{n-1}|c)p(\mathbf{x}_n|c)\cdots p(O|c, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n-1}, \mathbf{x}_n) \tag{23}$$

$$= p(c)p(\mathbf{x}_1|c)p(\mathbf{x}_2|c)\cdots p(\mathbf{x}_{n-1}|c)p(\mathbf{x}_n|c) \tag{24}$$

$$= p(c)p_c(\mathbf{x}_1)p_c(\mathbf{x}_2)\cdots p_c(\mathbf{x}_{n-1})p_c(\mathbf{x}_n). \tag{25}$$

We can simplify (22) and (23) because of the assumption that all members in $\mathbf{X}$ are drawn independently. Since O is

[1] A feature vector may depend on other feature vectors in real situations if they come from the same class.

TABLE I
TEST DATA SETS

DATA SET	CLASS	NUMBER OF VARIABLES		DATA
	NUMBER	CONTINUOUS	DISCRETE	SIZE
AUSTRALIAN	2	6	8	690
BREAST	2	30	0	569
CHESS	2	0	35	3196
CRX	2	6	9	690
GERMAN	2	24	0	1000
GLASS	7	9	0	214
HEART	2	5	8	270
HEPATITIS	2	6	13	155
IRIS	3	4	0	150
LETTER	26	16	0	20000
PIMA	2	8	0	768
SONAR	2	60	0	208
VEHICLE	4	18	0	846
WAVEFORM	3	21	0	5000
WINE	3	13	0	178

TABLE II
AVERAGE ACCURACIES (TOP) AND STANDARD DEVIATIONS (BOTTOM) USING DIFFERENT NAIVE BAYES CLASSIFICATION METHODS WHEN THE SIZE OF A HOMOLOGOUS SET IS TWO

DATA SET	HNB	VOTING	AVG	LAVG	MAX	LMAX	SNB
AUSTRALIAN	**94.74**	85.44	89.04	90.59	86.40	87.14	85.37
BREAST	**98.92**	94.87	94.21	97.67	94.82	94.46	95.96
CHESS	**91.84**	88.19	90.88	89.87	89.49	86.67	87.68
CRX	**93.62**	85.80	90.43	92.39	88.91	87.93	86.74
GERMAN	**81.72**	76.01	77.22	76.51	76.87	73.79	76.52
GLASS	**73.91**	61.74	73.04	70.43	71.74	63.48	63.48
HEART	**90.74**	83.53	86.03	88.09	85.15	85.73	82.71
HEPATITIS	**93.33**	84.67	91.33	89.33	91.21	90.00	86.33
IRIS	**100.0**	98.00	**100.0**	98.00	98.0	**100.0**	97.11
LETTER	**91.80**	73.57	86.53	84.75	85.26	84.02	73.07
PIMA	**83.25**	77.13	81.93	77.67	79.47	75.53	76.31
SONAR	**91.50**	79.00	78.25	86.50	77.75	79.50	80.23
VEHICLE	**65.56**	60.97	61.83	63.17	60.48	60.97	61.05
WAVEFORM	**88.42**	72.10	79.31	83.78	79.03	78.87	72.59
WINE	98.75	96.25	98.33	**99.16**	98.33	98.75	97.31
AVERAGE	89.21	81.15	85.24	85.86	84.19	83.12	81.45
BEST	14	0	1	1	0	1	0

DATA SET	HNB	VOTING	AVG	LAVG	MAX	LMAX	SNB
AUSTRALIAN	2.49	1.67	3.11	2.07	3.97	3.78	3.01
BREAST	1.74	2.32	1.55	2.08	2.78	2.14	1.71
CHESS	1.16	1.81	1.30	2.00	1.54	1.50	1.17
CRX	3.44	3.16	2.91	4.37	3.57	4.33	2.59
GERMAN	2.24	1.91	2.79	0.19	3.10	1.98	2.40
GLASS	6.45	6.68	7.48	5.77	7.08	3.98	6.79
HEART	3.01	3.27	5.01	5.36	2.82	3.89	3.09
HEPATITIS	5.96	5.81	5.21	6.80	4.34	5.37	6.05
IRIS	0.00	3.15	0.00	6.00	6.00	0.00	4.46
LETTER	0.50	0.76	0.42	0.40	0.65	0.37	0.52
PIMA	2.44	3.04	2.24	2.55	2.82	2.82	3.56
SONAR	5.61	7.34	6.52	6.04	6.84	5.22	6.21
VEHICLE	3.46	2.11	4.33	5.11	3.78	1.80	1.62
WAVEFORM	1.29	1.51	1.49	1.31	1.50	1.50	1.94
WINE	1.90	2.91	2.76	1.66	2.04	2.66	2.46

logically implied by the event that $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are of class c, $p(O|c, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n-1}, \mathbf{x}_n) = 1$ and we can reduce (23) and (24). For the sake of being concise, we rewrite (24) as (25). Finally, since the naive Bayes classifier assumes that all features are independent given class c, (25) can be decomposed by the following equations:

$$\begin{aligned}(25) = & p(c)p_c(x_{11}, x_{12}, \ldots, x_{1r})p_c(x_{21}, x_{22}, \ldots, x_{2r}) \\ & \cdots p_c(x_{n1}, x_{n2}, \ldots, x_{nr}) \qquad (26)\\ = & p(c)p_c(x_{11})p_c(x_{12})\cdots p_c(x_{1r})p_c(x_{21})\cdots \\ & p_c\left(x_{n(r-1)}\right)p_c(x_{nr}) \qquad (27)\\ = & p(c)\prod_{t=1}^{n}\prod_{l=1}^{r}p_c(x_{tl}). \qquad (28)\end{aligned}$$

Therefore, the classification rule is to pick class c that maximizes (28). We call this method *homologous naive Bayes (HNB)*. Note that the training procedure of the above methods remains unchanged as the standard naive Bayes classifier described in Section II.

IV. EMPIRICAL RESULTS

To evaluate the methods described in Section III, we selected several data sets from UCI ML repository [16] for our experiments. Results obtained with these data sets are widely reported in the machine learning literature. Table I lists information about each set.

A. Classifying Small Homologous Sets

In the first experiment, we investigated the performance of the methods when there are two or three elements in a homologous set. We first partitioned the data by class. Then, each class group was divided into five parts for running fivefold cross validation [17]. In each test set, we randomly drew two or three test examples to form a homologous set in our test data. Then, we trained the naive Bayes classifier with the training sets and used the six different methods to classify the test data for each data set.

We report the average and standard deviations of the accuracies after running fivefold cross validation on each data set. Table II gives the results of classifying homologous sets with two query vectors and Table III gives the results of classifying homologous sets with three query vectors. Located at the top of these two tables are the accuracies achieved by different methods, whereas at the bottom are the standard deviations of the accuracies. The results in Table II reveal that HNB outperforms the other methods in all data sets except one

TABLE III
AVERAGE ACCURACIES (TOP) AND STANDARD DEVIATIONS (BOTTOM) USING DIFFERENT NAIVE BAYES CLASSIFICATION METHODS WHEN THE SIZE OF A HOMOLOGOUS SET IS THREE

DATA SET	HNB	VOTING	AVG	LAVG	MAX	LMAX
AUSTRALIAN	**97.16**	94.40	90.52	93.88	87.54	78.58
BREAST	**99.27**	98.42	92.91	98.36	96.36	93.27
CHESS	**94.19**	93.09	90.14	90.38	90.33	85.05
CRX	**95.00**	92.61	90.43	92.75	89.57	91.16
GERMAN	**86.02**	81.88	78.01	76.22	77.60	71.79
GLASS	**80.71**	68.29	80.71	70.00	80.71	65.41
HEART	**95.00**	93.03	86.97	89.09	85.60	85.61
HEPATITIS	**95.00**	86.33	92.00	94.00	92.00	91.00
IRIS	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
LETTER	**96.85**	86.42	90.57	91.24	87.81	88.03
PIMA	**88.39**	85.74	83.71	78.99	78.37	74.32
SONAR	**93.95**	82.89	86.05	91.05	82.10	75.52
VEHICLE	**73.44**	66.75	64.25	71.25	66.50	60.12
WAVEFORM	**94.61**	77.71	82.14	87.45	80.96	79.42
WINE	**100.0**	98.63	99.09	99.09	98.63	98.63
AVERAGE	92.64	87.07	87.67	88.25	86.27	85.53
BEST	15	1	1	1	1	1

DATA SET	HNB	VOTING	AVG	LAVG	MAX	LMAX
AUSTRALIAN	3.38	3.37	3.21	2.68	2.85	4.10
BREAST	1.45	1.28	2.32	1.06	2.29	2.19
CHESS	1.19	1.62	1.79	1.87	2.52	1.96
CRX	2.76	2.78	3.79	2.34	3.03	2.93
GERMAN	3.07	3.26	2.56	2.51	2.36	1.38
GLASS	6.48	6.28	5.58	4.44	5.58	6.14
HEART	2.44	3.33	4.07	3.76	4.07	3.50
HEPATITIS	6.71	6.05	6.00	8.00	6.00	8.31
IRIS	0.00	0.00	0.00	0.00	0.00	0.00
LETTER	0.47	0.61	0.75	0.61	0.79	0.50
PIMA	3.67	3.65	2.20	3.61	2.58	4.29
SONAR	6.00	6.85	6.23	5.15	4.96	7.25
VEHICLE	5.21	3.75	4.51	5.00	4.99	5.28
WAVEFORM	1.27	1.62	2.13	1.35	0.84	0.98
WINE	0.00	2.91	2.72	2.72	2.08	2.91

(Wine, "LAvg"). In this table, we also list the accuracies of the standard naive Bayes classifier (SNB), which classifies one query vector at a time. When comparing the effectiveness of the standard naive Bayes classifier with others, we see that HNB achieves remarkable improvement. In contrast, the improvement by other methods is not only minor, but in some cases, the performance is worse than the SNB.

Let the accuracy of a classifier given a query vector be σ. Suppose there are two query vectors with the same unknown class label and the voting method is applied to classify them. In our implementation of voting, if the two results disagree, we randomly pick one as the final result. In this case, the expected accuracy is 0.5 when one of the results is correct. Therefore, the expected accuracy of the classifier when using voting is

$$\begin{aligned} E_{vote} &= \binom{2}{2} \sigma * \sigma + \binom{2}{1} \tfrac{1}{2} \sigma(1-\sigma) \\ &= \sigma^2 + \sigma(1-\sigma) \\ &= \sigma. \end{aligned}$$

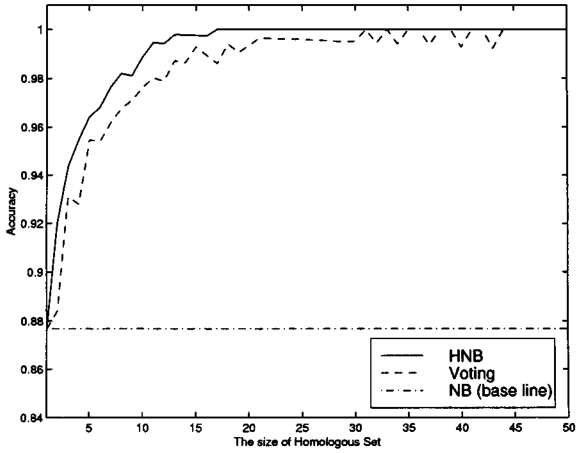

Fig. 2. Accuracies in classifying homologous sets using different sizes in the data set "Chess."

This shows that when we only know of two query vectors having the same unknown class label (i.e., the size of a homologous set is two), the expected accuracy will not be improved by voting. The experimental results given in Table II match our analysis. The accuracies of the two methods (SNB and voting) in most data sets are close.

In Table III, we grouped three query vectors in a homologous set. The results show that though the performance of voting is improved, the performance of HNB is improved even more and becomes the best of all methods for all data sets. In the next subsection, we discuss how the performance of HNB scales to increasing sizes of homologous sets.

B. Classifying Large Homologous Sets

A large homologous set increases the chances that a naive Bayes classifier correctly classifies a majority of the query vectors, thus ameliorating the performance of voting. In order to compare the performance of HNB and voting with different sizes of homologous sets, we selected three large data sets ("Chess," "Letter," and "Waveform," see Table I for their information) and repeated the same procedure as in the first experiment for different sizes of homologous sets (from 1 to 50 or 100). Note that when the size of a homologous set is 1, both HNB and voting are reduced to an SNB. This case serves as the baseline for performance evaluation.

Figs. 2–4 plot the resulting curves, which show that the performance of voting improves as the size of homologous sets increases, but the curves of HNB grow faster. HNB reaches perfect accuracy with less than 20 query vectors in the homologous sets. In contrast, voting requires many more vectors in order to reach the same performance.

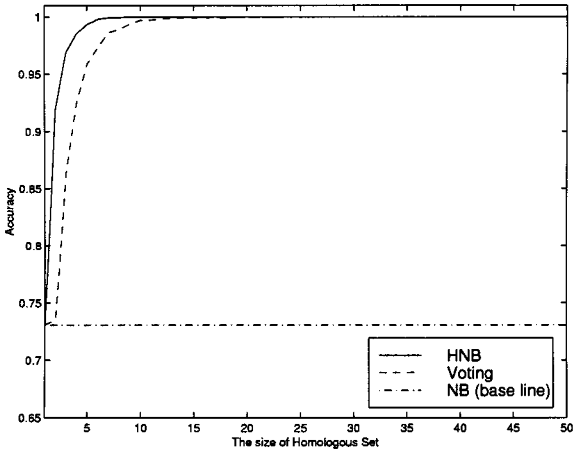

Fig. 3. Accuracies in classifying homologous sets using different sizes in the data set "Letter."

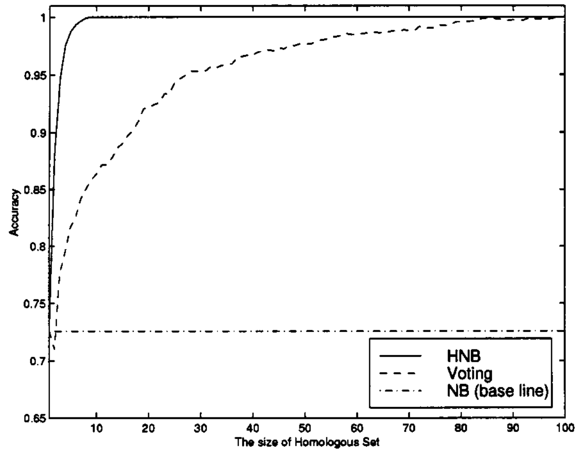

Fig. 4. Accuracies in classifying homologous sets using different sizes in the data set "Waveform."

V. Application in Speaker Recognition

Speaker recognition [15] is the process of automatically recognizing who is speaking on the basis of information obtained from speech waves. HNB is ideal for this task because we are easily able to sample a set of query vectors from the same unknown speaker. Since a large number of query vectors can be extracted from a short sentence, and obviously those vectors come from the same speaker, a speaker recognition system should take advantage of this information. Speaker recognition can be divided into two categories: "open-set" and "close-set." In "open-set" situations, the test speaker may not be registered and the system must identify that speaker as "unknown," while in "close-set" situations, the test speaker must be in the set of registered speakers. Speaker recognition can also be categorized according to its text dependency [18]. In text-dependent cases, speakers provide utterances of the same text for both training and testing. On the other hand, text-independent can use same or different text for training and testing. In this paper, we applied our work to a close-set, text-independent speaker recognition task.

A. Experiments

In this section, we compared HNB with voting in a speaker recognition task. The database used for the experiments reported in this paper is a subset of the TCC-300, a speech database in Mandarin Chinese maintained by many research institutes in Taiwan. We used the speech data recorded at the National Chiao-Tung University. All speech signals were digitally recorded in a laboratory using a personal computer with a 16-bit sound blaster card and a headset microphone. The sampling rate was 16 kHz. A 30-ms Hamming window was applied to the speech every 10 ms, allowing us to obtain 100 feature vectors from 1-s speech data. For each speech frame, both a twelfth-order linear predictive analysis and a log energy analysis were performed. We filtered the feature vectors whose log energy values were lower than 30. This can be viewed as a silence-removing process. A feature vector for training or testing contained the 12 linear predictive parameters. More than ten sentences were recorded from each subject speaker, and from each sentence, more than 4000 feature vectors were extracted.

TABLE IV
Accuracies of the Application to Speaker Recognition

	NUMBER OF SPEAKERS			
Size of	2		5	
homologous set	HNB	Voting	HNB	Voting
2	90.34±1.38	80.02±2.24	70.97±1.72	58.77±2.00
5	98.70±0.43	94.75±1.89	89.32±1.35	78.18±2.36
10	100.0±0.00	97.80±1.56	97.28±0.67	90.44±1.47
20	100.0±0.00	100.0±0.00	99.68±0.29	97.04±1.79
50	100.0±0.00	100.0±0.00	100.0±0.00	99.40±0.48
100	100.0±0.00	100.0±0.00	100.0±0.00	100.0±0.00

	NUMBER OF SPEAKERS			
Size of	10		15	
homologous set	HNB	Voting	HNB	Voting
2	50.29±1.91	37.97±1.42	42.68±1.57	30.37±0.79
5	72.91±2.78	54.62±2.31	65.50±2.26	44.91±0.15
10	87.00±2.30	67.40±2.47	81.84±2.43	57.65±1.11
20	96.04±1.98	78.92±2.28	93.47±2.34	70.32±1.32
50	99.40±0.96	87.40±3.15	98.20±1.68	79.46±2.27
100	100.0±0.00	88.00±4.60	99.33±0.84	82.80±2.77

The experimental procedure was conducted as follows. We randomly selected five sentences for each speaker, one for testing and the others for training. Then, we randomly selected 1000 feature vectors from each sentence. Hence, for each speaker there were 4000 feature vectors for training and 1000 feature vectors for testing. We ran fivefold cross validation on different numbers of speakers and different sizes of homologous sets. We report the average and standard deviations of the accuracies in Table IV. The accuracies for both methods (HNB and voting) improve significantly in all cases when the sizes of homologous sets increase. However, HNB reaches high accuracy faster than voting. This is consistent with the results in Figs. 2–4.

B. Gaussian Mixture Model (GMM)

The most common and successful approaches to close-set, text-independent speaker recognition include the Gaussian mixture model approach (GMM) [19] and the hidden Markov model approach (HMM) [20]. In recent speaker recognition evaluations carried out by the National Institute of Standards and Technology (NIST), the best GMM-based systems have outperformed the HMM-based systems [21]. In this section, we will briefly review the GMM approach to speaker recognition. Then, we will empirically compare HNB and GMM in Section V-C.

A Gaussian mixture density is a weighted sum of M component densities and is given by the form [19]

$$p(w|\lambda) = \sum_{i=1}^{M} c_i b_i(w)$$

where w is a d-dimensional random vector, $b_i(w)$, $i = 1, \ldots, M$ is the component density and c_i, $i = 1, \ldots, M$ is the mixture weight. Each component density is a d-variate Gaussian function of the form

$$b_i(w) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left\{-\tfrac{1}{2}(w-\mu_i)'\Sigma_i^{-1}(w-\mu_i)\right\}$$

with mean vector μ_i and covariance matrix Σ_i. The mixture weights must satisfy the constraint that

$$\sum_{i=1}^{M} c_i = 1.$$

The complete Gaussian mixture density is parameterized by the mean vectors, covariance matrices, and mixture weights from all component densities. These parameters are collectively represented by the notation

$$\lambda = \{c_i, \mu_i, \Sigma_i\}, \qquad i = 1, \ldots, M.$$

Then, each speaker is represented by a GMM and is referred to by his/her model parameter set λ. A GMM parameter set λ for a speaker is estimated using the standard expectation maximization (EM) algorithm [22]. For a sequence of T query vectors $X = \{x_1, x_2, \ldots, x_T\}$, the GMM log-likelihood can be written as

$$p(X|\lambda) = \sum_{t=1}^{T} \log p(x_t|\lambda).$$

In the standard identification approach, the test speaker i^* is recognized from a set of S speakers by

$$i^* = \arg \max_{i=1}^{S} p(X|\lambda_i).$$

Several system parameters must be tuned for training a Gaussian mixture speaker model, but a good theoretical guide for setting the initial values of those parameters has not been found. The most critical parameter is the order M of the mixture. Choosing too few mixture components can produce a speaker model that cannot accurately model the distinguishing characteristics of a speaker's distribution. Choosing too many components reduces performance when there are a large number of model parameters relative to the available training data [19].

TABLE V
ACCURACIES OF HNB AND GMM FOR RECOGNIZING 16 (TOP) AND 30 (BOTTOM) SPEAKERS

SIZE OF	ACCURACY FOR 16 SPEAKERS	
HOMOLOGOUS SET	HNB	GMM
1	31.43±1.09	52.31±1.51
20	92.90±2.93	99.42±0.78
50	97.93±2.25	100.0±0.00
100	99.13±1.22	100.0±0.00
200	99.50±1.00	100.0±0.00
300	99.58±0.83	100.0±0.00
400	100.0±0.00	100.0±0.00
500	100.0±0.00	100.0±0.00
600 (6 SECS)	100.0±0.00	100.0±0.00

SIZE OF	ACCURACY FOR 30 SPEAKERS	
HOMOLOGOUS SET	HNB	GMM
1	24.31±0.77	44.23±1.46
20	94.43±2.29	99.68±0.44
50	98.93±1.03	99.93±0.13
100	99.53±0.65	100.0±0.00
200	99.60±0.53	100.0±0.00
300	99.78±0.44	100.0±0.00
400	99.85±0.66	100.0±0.00
500	100.0±0.00	100.0±0.00
600 (6 SECS)	100.0±0.00	100.0±0.00

TABLE VI
AVERAGE TRAINING TIME OF HNB AND GMM FOR DIFFERENT NUMBER OF SPEAKERS

NUMBER OF	AVERAGE TRAINING TIME		
SPEAKERS	HNB	GMM	SPEEDUP
16	382.22	128056.44	335.03
30	716.22	240300.22	335.51

C. Comparisons Between HNB and GMM

In this section, HNB is empirically compared with the most successful statistical model (GMM) for speaker recognition. The experimental procedure is the same as in Section V-A and the same data set is used. $M = 32$ for GMM is used in our experiments. This setting is suggested in [19] and works well in our data sets. We also ran fivefold cross validation on two different numbers of speakers and different sizes of homologous sets. One hundred test data represents 1 s of speech data. We reported the average accuracies and their standard deviation. We also reported the average CPU time ticks taken for training and classification. Table V shows the results of recognition accuracies for 16 and 30 speakers. Tables VI and VII show the average training and classification time (the size of homologous is from 1 to 600). The experiments were run on Pentium III 800 MHz PCs with 256M DRAM.

TABLE VII
TESTING TIME OF HNB AND GMM FOR THE TEST ON 16 (TOP) AND 30 (BOTTOM) SPEAKERS

SIZE OF	TEST TIME FOR 16 SPEAKERS (PER QUERY)		
HOMOLOGOUS SET	HNB	GMM	SPEEDUP
1	0.197	3.880	19.70
20	3.570	50.928	14.27
50	8.718	122.956	14.10
100	16.900	242.500	14.34
200	32.800	485.125	14.79
300	48.541	728.042	15.00
400	64.500	967.395	15.00
500	80.000	1223.648	15.30
600 (6 SECS)	95.500	1465.250	15.34
AVERAGE	–	–	15.31

SIZE OF	TEST TIME FOR 30 SPEAKERS (PER QUERY)		
HOMOLOGOUS SET	HNB	GMM	SPEEDUP
1	0.378	5.867	15.52
20	6.842	103.966	15.20
50	16.403	251.153	15.31
100	31.700	500.313	15.78
200	61.533	992.093	16.13
300	90.911	1490.000	16.39
400	120.666	1983.466	16.44
500	149.933	2484.633	16.57
600 (6 SECS)	179.466	2983.466	16.00
AVERAGE	–	–	16.79

The results show that GMM requires smaller numbers of test vectors than HNB to reach a high accuracy. However, given more feature vectors in the homologous sets, HNB quickly catches up and reaches comparable accuracy to GMM. This is not surprising because the GMM is capable of modeling the correlations between the variables in a feature vector, but the correlations are disregarded in the naive Bayes classifier due to its independent assumption. As a result, when the size of a homologous set is 1, the accuracies of HNB will be much lower than GMM. However, HNB gains several advantages from the independent assumption. One is that the implementation of HNB is much simpler than the GMM algorithm. Other advantages of HNB include the faster training and classification speeds. In our experiment, the average training speed of HNB surpasses GMM by about 355 times, and the average classification speed by about 15 (see Tables VI and VII for training time and classification time, respectively). In addition, HNB reaches high accuracies (>99%) by using about 1 s of speech data. This shows that it is not necessary to use a very large number of query vectors in order to boost HNB's accuracies. An interesting phenomenon here is that, though more query vectors are required in order for HNB to reach perfect accuracy, HNB's classification speed is still faster than GMM's speed, as shown in Table VIII.

TABLE VIII
SIZE OF A HOMOLOGOUS SET AND CLASSIFICATION TIME FOR REACHING THE PERFECT ACCURACY

NUMBER OF	SIZE REQUIRED		TEST TIME		
SPEAKERS	HNB	GMM	HNB	GMM	SPEEDUP
16	400	50	64.50	122.956	1.91
30	500	100	149.933	500.000	3.35

Based on the discussion above, we draw the following conclusions on applying HNB in speaker recognition task. Since one additional second of speech is a low cost to the speakers, it is a tolerable or even favorable tradeoff in favor of HNB. Hence, if it is not too difficult to obtain speech data from the same speaker for classification, or if low-cost implementation is required, the HNB can be a useful approach.

VI. CONCLUSIONS

The naive Bayes classifier is widely used in many classification tasks because its performance is competitive with state-of-the-art classifiers, it is simple to implement, and it possesses fast execution speed. In this paper, we discussed the problem of how to classify a set of query vectors from the same unknown class with the naive Bayes classifier. We showed that a classifier must deliberately take advantage of the knowledge that all data have the same unknown class label; otherwise the knowledge will not improve the expected accuracy. Then, we proposed the method HNB and compared it with several simple methods (Avg, LAvg, Max, LMax, and Voting). The experimental results show that HNB can take advantage of the prior information that all members in a homologous set have the same class label, improve accuracy, and outperform the other methods when the naive Bayes model is used.

We also compared HNB and voting for the application of speaker recognition. Experimental results show that HNB can work well on this task and is more suitable than voting. Finally, HNB was compared with the GMM approach in speaker recognition. Experimental results reveal that, although HNB can reach the same level of accuracies as GMM by using about 1 s more of test speech data, HNB's execution speed is much faster than GMM, and HNB has low implementation cost. Hence, we suggest that HNB is useful in the domain of speaker recognition and may be applied to other applications, if homologous sets are available in that problem domain.

ACKNOWLEDGMENT

The authors wish to thank anonymous reviewers for their valuable comments.

REFERENCES

[1] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, New York: Academic, 1990.

[2] C.-H. Lee, F. K. Soong, and K. K. Paliwal, *Automatic Speech and Speaker Recognition*. Norwell, MA: Kluwer, 1996.

[3] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann, 1993.

[4] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Mach. Learn.*, vol. 29, pp. 131–163, 1997.

[5] I. Parsa. (1997) KDD-CUP 1997 presentation. [Online]. Available: http://www.ncst.ernet.in/kbcs/vivek/issues/10.4/kdd/kdd.html.

[6] P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss," *Mach. Learn.*, vol. 29, pp. 103–130, 1997.

[7] T. M. Mitchell, *Machine Learning*, New York: McGraw-Hill, 1997.

[8] G. John and P. Langley, "Estimating continuous distributions in Bayesian classifiers," in *Proc. 11th Annu. Conf. UAI*, 1995, pp. 338–345.

[9] S. S. Wilks, *Mathematical Statistics*, New York: Wiley, 1962.

[10] R. Almond, *Graphical Belief Modeling*. London, U.K.: Chapman & Hall, 1995.

[11] B. Cestnik and I. Bratko, "On estimating probabilities in tree pruning," in *Machine Learning—EWSL-91, European Working Session on Learning*. Berlin: Springer-Verlag, 1991, pp. 138–150.

[12] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous valued attributes for classification learning," in *Proc. 13th IJCAI*, 1993, pp. 1022–1027.

[13] J. Dougherty, R. Kohavi, and M. Sahami, "Supervised and unsupervised discretization of continuous features," in *Machine Learning: Proceedings of the 12th International Conference (ML '95)*. San Mateo, CA: Morgan Kaufmann, 1995.

[14] C.-N. Hsu, H.-J. Huang, and T.-T. Wong, "Why discretization works for naive Bayesian classifiers," in *Machine Learning: Proceedings of the 17th International Conference (ML 2000)*. San Mateo, CA: Morgan Kaufmann, 2000.

[15] S. Ross, *A First Course in Probability*. Englewood Cliffs, NJ: Prentice-Hall, 1998.

[16] C. Blake and C. Merz. (1998) UCI repository of machine learning databases. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html.

[17] M. Stone, "Cross-validatory choice and assessment of statistical preditions," *J. R. Stat. Soc.*, vol. 36, pp. 111–147, 1974.

[18] H. Gish and M. Schmidt, "Text-independent speaker identification," *IEEE Signal Processing Mag.*, vol. 11, pp. 18–32, 1996.

[19] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. Speech Audio Processing*, vol. 3, pp. 72–83, Jan. 1995.

[20] J. de Veth and H. Bourlard, "Comparision of hidden Markov model techniques for automatic speaker verification in real-world conditions," *Speech Commun.*, vol. 17, pp. 81–90, 1995.

[21] M. A. Przybocki and A. F. Martin, "NIST speaker recognition evaluation," in *Workshop Speaker Recognition and Its Commercial and Forensic Applications (RLA2C)*, Avignon, France, 1998.

[22] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *J. R. Stat. Soc.*, vol. Series B, no. 39, pp. 1–38, 1977.

Hung-Ju Huang was born on August 1, 1974, in Taipei, Taiwan, R.O.C. He received the B.S. degree in computer science and information engineering from Ta-Tung Institute of Technology, Taipei, Taiwan, in 1996. Currently, he is pursuing the Ph.D. degree in the Department of Computer and Information Science, National Chiao-Tung University, Hsinchu, Taiwan. His main research interests include maching learning, artificial intelligence, and VLSI CAD.

Chun-Nan Hsu (M'01) received the B.S. degree in computer engineering from National Chiao-Tung University (NCTU), Hsinchu, Taiwan, R.O.C., in 1988, and the M.S. and Ph.D. degrees in computer science from the University of Southern California, Los Angeles, in 1992 and 1996, respectively.

Currently, he is an Assistant Research Fellow at the Institute of Information Science, Academia Sinica, Taipei, Taiwan. He was Assistant Professor in the Department of Computer Science and Engineering at Arizona State University, Tempe, from 1996 to 1998, and Adjunct Assistant Professor in the Department of Computer Science and Information Engineering at NCTU from 1999 to 2000. His current research interests include machine learning, knowledge discovery and data mining, databases, and intelligent Internet agents and their applications in bioinformatics. He has two U.S. patents pending.

Dr. Hsu is a member of ACM and AAAI. He has served as the Secretariat General of Taiwanese Association for Artificial Intelligence (TAAI) since 2001. He is on the Program Committee of the 1998 National Artificial Intelligence Conference (AAAI-98).