Internet Traffic Classification Using Hidden Naive Bayes Model

Fatemeh Ghofrani, Azizollah Jamshidi, Alireza Keshavarz-Haddad
School of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran

Email: {f.ghofrani, jamshidi, keshavarz}@shirazu.ac.ir

***Abstract*—Internet traffic classification plays an important role for network management. In fact, operators need to better predict future traffic behavior to identify anomalous situations. We present here an approach for traffic classification using Hidden Naive Bayes model and a supervised discretization scheme. This approach can achieve an appropriate performance on a range of application types with accessing only the information that remains unchanged after encryption. At first, we use a supervised method based on idea behind Holte's 1R algorithm for discretization of continuous features derived from packet headers. Then, in order to assign flows to their respective classes, we utilize Hidden Naive Bayes (HNB) model. Finally, we test our scheme using a subset of two data sets and compare it to Tree-Augmented Naive Bayes (TAN) algorithm. Various performance measures namely Accuracy (Auc) and Trust are used for quantitative analysis of our results. Experimental results reveal that our proposed modeling approach based on HNB not only achieves a higher performance in terms of both measures in comparison to TAN algorithm but also learns very well even with a small number of training flows.**

I. Introduction

In order to solve issues related to network management, traffic classification plays an important role for internet service providers (ISPs). In fact, operators need to know what applications are flowing on their networks so they can support their various goals. Moreover, one of the most important parts of intrusion detection is traffic classification. Thus, there is a great need for powerful and reliable traffic classification.

Commonly, IP traffic classification techniques are based on inspecting the content of each packet in some points of the network. Consecutive IP packets having the same 5-tuple of characteristics such as protocol type, source address, source port, destination address and destination port are considered as a flow that we wish to determine its application [1].

In simple classification methods, it is assumed that most applications are identified using the well known TCP or UDP port numbers. However, nowadays having extremely fast growth in uses of unpredictable port numbers, many classification methods attempt to identify application by seeking a well known protocol signature within the TCP or UDP payloads. Unfortunately, the techniques based on deep packet inspection are not usual. Because, firstly costumers may use the encryption methods and therefore actual content of the packets is not be available. Secondly, inspecting the content of the payload in order to interpret flows has extremely heavy load. Recently, new methods infer applications using reorganization of the statistical patterns within flows. Their final goal is classifying IP traffic flows into groups that have similar traffic patterns.

In recent years, utilizing the statistical methods is proposed to classify the network traffic. These methods typically extract some features according to the characteristics of network traffic flows or individual packets. For example, the average packet length or size of the first packet of each connection are examples of these features. In order to classify, a variety of methods including supervised classification, unsupervised clustering, semi-supervised classification and methods based on Markov models have been proposed.

Several network traffic classification schemes are presented in the previous literatures [2]-[10]. For example, in [2] three unsupervised clustering algorithms, namely K-Means, DBSCAN and AutoClass are considered. The experimental results show that both K-Means and DBSCAN work very well and much more quickly than AutoClass. Moreover, their results show that although DBSCAN has lower accuracy compared to K-Means and AutoClass, DBSCAN produces better clusters. In [3], a semi-supervised classification method that can adapt to both known and unknown applications is presented. This method allows classifiers to be designed from training data that consists of only a few labeled and many unlabeled flows. In [4], a technique to identify the applications of internet traffic is presented. This technique relies on the monitoring of the first five packets of a TCP connection and K-Means algorithm. This work shows that the size of the first few packets is a good predictor of the application related with a flow, because it captures the applications negotiation phase, which is different among the applications. In [5], a supervised Naive Bayes technique to classify internet traffic is proposed. In this work, 248 features based flow are utilized to train the classifier. Moreover, two improvements for the classifier with the use of Naive Bayes Kernel Estimation (NBKE) and Fast Correlation-Based Filter (FCBF) methods have been achieved. These improvements help to decrease the feature space and make better the classifier performance. This work with the application of Bayesian neural network approach is extended in [6]. It has been revealed that Bayesian neural network achieves very good performance compare to Naive Bayes technique. In [7], the performance of five algorithms including Naive Bayes with Discretization (NBD), Naive Bayes with Kernel Density Estimation (NBK) , C4.5 Decision Tree, Bayesian Network and Naive Bayes Tree using Weka for traffic classification

978-1-4799-1972-7/15/$31.00

is investigated. The results demonstrate that most algorithms reach high flow accuracy. Moreover, algorithms sorted from highest to lowest order in terms of classification speeds are: C4.5, NBD, Bayesian Network, Naive Bayes Tree, NBK. Also, algorithms sorted from highest to lowest order in terms of modeling time are: Naive Bayes Tree, C4.5, Bayesian Network, NBD, NBK.

In this paper, the proposed modeling approach relies on a Hidden Naive Bayes (HNB) [11] on the basis of a more general Bayesian Network approach that has revealed its remarkable performance in classification. In our work, we evaluate the effectiveness of HNB model under a supervised discretization method for classification of internet traffic applications. Using training data, a structure of HNB is created for those applications which we want to classify and distinguish. The classification of new flows is based on the maximum likelihood which selects the application which the model yields the highest a-posteriori probability for the given features. We evaluate our approach using the data set described in [12]. Different performance measures namely Accuracy (Auc) and Trust are used for quantitative analysis of our results. Experimental results show that our proposed modeling approach based on HNB provides a minimum improvement 12% in both measures compared to Tree Augmented Naive Bayes [14] (TAN) algorithm even under the situation of very few training flows.

The rest of the paper is organized as follows: In section II, we describe our methodology for modeling and classification of internet traffic. Then, experimental results are presented for evaluation of our scheme in section III. Finally, section IV concludes our paper.

II. Methodology

The proposed method involves two stages: 1- supervised discretization of continuous features 2-modeling and classification process. Next, these stages are described in detail.

A. Supervised discretization of continuous features

Many algorithms expanded in the field of machine learning focus on learning in nominal (categorical) feature spaces. However, there are many real-world classification algorithms that include continuous (numerical) features where such algorithms could not be utilized unless the continuous features are first discretized [15].

In our study, a supervised method based on idea behind Holte's 1R algorithm [16] for discretization of continuous features is utilized. This method is named as supervised discretization, because it utilizes the class labels across discretization process. In this method, after sorting the continuous features, a number of intervals are formed by placing some cut points between every pair of different values. Then, predicted class of each interval is determined based on majority of labels in that interval. In the next stage, cut points between intervals that predict the same class are removed. Finally, each interval represents a bin and discretization method completes. The following simple example can clarify this algorithm. In the example, the first row is the values of a feature after sorting, while the second row shows the class label (Y or N).

10	10	10	12	12	15	17	17	17	20	23
Y	Y	N	N	N	N	Y	Y	N	Y	N

Placing some cut points between every pair of different values that forms a number of intervals gives:

10	10	10	12	12	15	17	17	17	20	23
	Y		N		N		Y		Y	N

The cut point between 12 and 15 also 17 and 20 can now be removed without changing of class labels.

10	10	10	12	12	15	17	17	17	20	23
	Y			N				Y		N

Finally, each interval represents a bin across discretization process.

B. Modeling and classification process

One of the most effective classifiers, in the sense that its predictive performance is competitive with state-of-the-art classifiers, is the so-called naive Bayesian classifier [14]. This classifier computes the conditional probability of each attribute A_i given the class label C using training data. Then classification process is performed by utilizing Bayes rule to calculate the probability of C given the particular features of $A_1; ...; A_n$, and then determining the class with the highest posterior probability. In this computation, a strong independence assumption is considered as all the features A_i

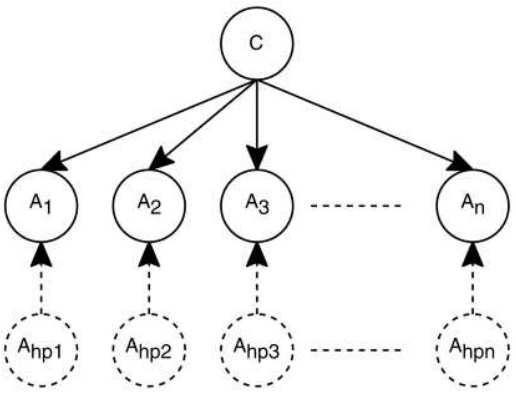

Fig. 1. The HNB structure [11].

knowing the value of the class C are conditionally independent. However, this assumption is often violated in the real data. The Bayesian network is a generalization of Naive Bayes and it solves the problem of independency of the features. On the other hand, although a Bayesian network can represent arbitrary feature dependencies, it is intractable to learn it from data [17]. Thus restricted Bayesian networks such as TAN, are more practical. However, in TAN scheme, despite of the fact that there may be several features with similar influence, each feature is allowed to depend on at most one feature. Hidden Nave Bayes is a model that can address the problem of the intractable computational complexity in learning phase and also can take into account the influence of all features using a hidden parent. Fig. 1 shows the structure of an HNB. In this figure, class node C is the parent of all feature nodes. Each feature A_i has a hidden parent $A_{\mathrm{hp}i}; i = 1; 2; ...; n$, shown by a dashed circle. The edge from the hidden parent $A_{\mathrm{hp}i}$ to A_i is also shown by a dashed directed line, to differentiate it from the ordinary edges.

The Joint distribution that describes a HNB model is defined as follows:

$$P(A_1, ..., A_n, C) = P(C) \prod_{i=1}^{n} P(A_i|A_{\mathrm{hp}i}, C), \quad (1)$$

where

$$P(A_i|A_{\mathrm{hp}i}, C) = \sum_{j=1, j\neq i}^{n} W_{ij} * P(A_i|A_j, C). \quad (2)$$

Eq.2 shows that the hidden parent $A_{\mathrm{hp}i}$ for the feature A_i is calculated using a combination of the weighted influences from all other features.
In HNB model, class of a test flow $E = (a_1, a_2, \ldots, a_n)$, where a_i is the value of ith feature A_i, is calculated as follows:

$$c(E) = arg \max_{c\in C} P(c) \prod_{i=1}^{n} P(a_i|a_{hp_i}, c), \quad (3)$$

where

$$P(a_i|a_{hpi}, c) = \sum_{j=1, j\neq i}^{n} W_{ij} P(a_i|a_j, c). \quad (4)$$

In a HNB, dependency between features are represented by hidden parents of features. In Eq. 2, $P(A_i|A_j, C)$ are employed to define hidden parents. It should be noted that TAN scheme allows to utilize only one feature parent and therefore the influences from other features have to be eliminated. In a HNB, the influences from all other features can be considered and a weight is utilized to represent the importance of each feature in hidden parent. There are two general approaches to calculate weights: performing a cross-validation-based search, or directly computing the estimated values from data [11]. We apply the latter, and use the conditional mutual information between two features A_i and A_j as the weight of $P(A_i|A_j, C)$. Therefore, W_{ij} is defined as:

$$W_{ij} = \frac{I_p(A_i; A_j|C)}{\sum_{j=1, j\neq i}^{n} I_p(A_i; A_j|C)}, \quad (5)$$

where

$$I_p(A_i; A_j|C) = \sum_{a_i, a_j, c} P(a_i, a_j, c) \log \frac{P(a_i, a_j|c)}{P(a_i|c)P(a_j|c)}. \quad (6)$$

In the learning phase of a HNB model, training data are utilized to estimate the parameters in the model. Furthermore, probabilities $P(c)$ and $P(a_i|a_j, c)$ are defied using the M-estimation as:

$$P(c) = \frac{F(c) + \frac{1}{k}}{t + 1}, \quad (7)$$

$$P(a_i|a_j, c) = \frac{F(a_i, a_j, c) + \frac{1}{n_i}}{F(a_j, c) + 1}, \quad (8)$$

where $F()$ is the number of times that combination of terms is visited in the training data, k is the number of classes, t is the number of instances in the training data and n_i is the number of values of feature A_i.

III. Experimental Results

A. Datasets and Traffic Features

Experiments are implemented on a subset of data described in [12]-[13]. In order to evaluate the performance of classification method, we use two datasets which consist of TCP traffic only. One dataset called *wide* dataset contains 6 classes. The other called *isp* dataset is categorized into 11 classes. Details of selected classes in this study are listed in the Table I. We have taken 200 and 50 flows for each class as training instances and 800 flows for each class as test instances. Training and test instances in these subsets are selected in a random way.

In order to evaluate the performance of our scheme, two criteria including Accuracy (*Auc*) and *Trust* are utilized. In fact, *Auc* value describes the classification accuracy and *Trust* value is a metric of how much the classification can be reliable. These criteria are defined as follows:

$$Auc = \frac{TP}{TP + FN} \quad (9)$$

$$Trust = \frac{TP}{TP + FP} \quad (10)$$

Where we have:

TABLE I
Considered traffic in datasets [12]

Class Number	Applications in *isp*	Applications in *wide*
1	BT	P2P
2	DNS	DNS
3	FTP	FTP
4	HTTP	WWW
5	IMAP	CHAT
6	MSN	MAIL
7	POP3	-
8	SMTP	-
9	SSH	-
10	SSL	-
11	XMPP	-

- True Positive (TP)=The number of flows correctly classified as belonging to class X.
- True Negative (TN)=The number of flows correctly classified as not belonging to class X.
- False Positive (FP)=The number of flows incorrectly classified as belonging to class X.
- False Negative (FN)=The number of flows incorrectly classified as not belonging to class X.

In this work, six features to represent TCP flows are used. These features are listed as follows:

1) The number of packets transferred from client to server
2) Maximum packet size from client to server
3) Minimum packet size from client to server
4) Variance of packet size from client to server
5) The number of packets transferred from server to client
6) Maximum packet size from server to client

B. Results of Experiments

In order to verify capability of the work presented in this paper, classification of traffic flows using 1R discretization and TAN classifier is also implemented and evaluated. The results of these experiments are listed in Table II-V for *wide* and *isp* datasets. In order to get comparable estimates between TAN and HNB, the first experiment is implemented on 800 test instances using 200 training data. Table II and III illustrates the results of this experiment. From the results, it is observed that TAN classifier using 1R discretization scheme achieved the average *Auc* 65.56% and 68.91% on *isp* and *wide* dataset, respectively. Corresponding *Trust* values achieved on these datasets are 65.44% and 68.16% respectively. The HNB classifier provides an improvement of *Auc* values, i.e. 81.16% and 82.56% on *isp* and *wide* datasets, respectively. For *Trust* criteria, HNB reaches 81.78% and 84.23% on these datasets. Thus there is an increase of nearly 14% of *Auc* by HNB classifier, and the corresponding *Trust* value is increased by 16% compared to TAN classifier.

The second experiment is implemented on 800 test instances using 50 training flows. The classification results of this experiment is depicted in Table IV-V. It is observed that with 50 training flows, the performance of the HNB classifer is better compared with TAN classifier. For example, on *isp* dataset, the *Auc* values are 74.89% and 62.18% using the HNB scheme and TAN, respectively. Similarly, the corresponding *Trust* values for these classifiers are 75.16%, 61.80%. For *wide* dataset, nearly the same *Auc* value improvement is there as in the case of 50 training flows, i.e., with the HNB classifier and TAN method these values are 77.60% and 62.02%, respectively. Also there is an increment in *Trust* value of 17% with HNB classifier compared with TAN scheme.
In addition, the results also reveal that the *Auc* and *Trust* values obtained with HNB classifier for 50 training data is more than with TAN at 200 training data. This is particularly important in the situations where a small number of training flows are available.

C. Computational Complexity

In order to better understanding the limitations and strengths of our scheme, analysis of time and space complexity HNB and TAN schemes are represented in Table VI [18]. In this table, parameters are defined as follows:

- k is the number of classes.
- t is the number of training instances.
- n is the number of features.
- v is the mean number of mean values per feature values.

Space complexity of TAN at training time is equal to $O(k(nv)^2)$ since it makes a one-dimensional table of class probability estimates and a three-dimensional table including a conditional probability estimate for each feature-value, conditioned on each other feature-value and each class. As calculating each entry for every combination of the two feature values is required in every sample, the time complexity of making the three dimensional probability table is $O(tn^2)$. In order to form the conditional mutual information matrix, we need to consider for each pair of features of every pair wise combination of their respective values in conjunction with each class. Hence, the time complexity is equal to $O(k(nv)^2)$. By utilizing a maximal spanning tree that its time complexity is $O(n^2 \log n)$, the parent function is created. At

TABLE II
THE PERFORMANCE OF TAN CLASSIFIER USING 1R DISCRETIZATION SCHEME AND 200 TRAINING INSTANCES.

Class Number	Auc in *isp*	Trust in *isp*	Auc in *wide*	Trust in *wide*
1	77.125	60.125	60.875	57.907
2	86.625	81.388	89.875	83.217
3	67.750	66.097	49.500	53.80
4	31.625	32.857	46.00	56.790
5	49.625	52.862	96.875	93.26
6	47.375	58.759	70.375	63.977
7	74.500	73.853	-	-
8	70.000	72.916	-	-
9	96.750	96.750	-	-
10	39.500	38.964	-	-
11	80.375	85.052	-	-
Ave	**65.568**	**65.449**	**68.916**	**68.159**

TABLE III
THE PERFORMANCE OF HNB CLASSIFIER USING 1R DISCRETIZATION SCHEME AND 200 TRAINING INSTANCES.

Class Number	Auc in *isp*	Trust in *isp*	Auc in *wide*	Trust in *wide*
1	67.875	97.837	52.250	97.892
2	91.000	97.981	94.375	90.203
3	91.125	84.472	80.375	70.427
4	45.875	50.690	81.875	75.722
5	88.250	8.0410	99.875	96.960
6	66.500	71.505	86.625	74.197
7	92.500	87.886	-	-
8	92.000	84.378	-	-
9	99.750	97.317	-	-
10	64.175	53.661	-	-
11	93.250	93.483	-	-
Ave	**81.159**	**81.784**	**82.562**	**84.234**

TABLE IV
THE PERFORMANCE OF TAN CLASSIFIER USING 1R DISCRETIZATION SCHEME AND 50 TRAINING INSTANCES.

Class Number	Auc in *isp*	Trust in *isp*	Auc in *wide*	Trust in *wide*
1	75.750	53.345	51.750	54.545
2	81.375	65.100	76.875	67.286
3	68.000	64.839	45.125	52.623
4	21.500	25.671	38.250	39.843
5	50.625	50.185	97.250	91.421
6	44.875	47.051	62.875	61.192
7	63.50	69.209	-	-
8	70.000	80.229	-	-
9	98.000	96.790	-	-
10	27.875	35.794	-	-
11	82.500	91.666	-	-
Ave	**62.181**	**61.807**	**62.0208**	**61.152**

TABLE V
THE PERFORMANCE OF HNB CLASSIFIER USING 1R DISCRETIZATION SCHEME AND 50 TRAINING INSTANCES.

Class Number	Auc in *isp*	Trust in *isp*	Auc in *wide*	Trust in *wide*
1	67.50	96.601	51.750	92.00
2	86.000	92.473	96.625	81.282
3	81.125	74.171	69.750	72.093
4	30.500	42.657	66.875	68.943
5	69.875	72.976	98.375	92.588
6	62.875	61.341	82.125	65.865
7	87.875	79.524	-	-
8	91.500	81.333	-	-
9	99.750	93.442	-	-
10	55.250	44.691	-	-
11	91.625	87.574	-	-
Ave	**74.897**	**75.162**	**77.604**	**78.795**

the classification step, in order to determine the label of a test flow, time complexity $O(kn)$ is required. At training time, the three dimensional conditional probability table generated can be compressed by storing probability estimates for each feature-value given its parent and the class. Hence, the space complexity is $O(knv^2)$. HNB forms probability tables and conditional mutual information matrix as TAN does, at training time. Hence, space complexity is equal to $O(k(nv)^2)$. The time complexity of calculating weights is $O(n^2)$ and therefore, overall time complexity is $O(tn^2 + k(nv)^2)$. At classification step, we need the tables of probability estimates generated at training time of space complexity $O(k(nv)^2)$. Since, we require to consider each pair of appropriate parent and child feature within each class, the time complexity of labeling a single sample is $O(kn^2)$.

IV. CONCLUSION

In this paper, we proposed to apply HNB model for internet traffic using a supervised feature discretization scheme. HNB is a model that can address the problem of the intractable computational complexity in learning phase of Bayesian Networks and also can take into account the influence of all features using a hidden parent. To modeling of traffic data using HNB, we needed to nominal feature spaces. Hence, a supervised discretization scheme that utilizes class labels across discretization is employed. Our scheme for traffic classification was implemented and evaluated through some experiments. Classification results demonstrate that our proposed modeling approach based on HNB provides an improvement 12% of Auc and Trust measure compared to TAN algorithm for internet traffic classification. The HNB approach implemented in this work, forces the relations among features to be the same for all classes. We believe that using local networks for each application can improve the performance of HNB model for internet traffic classification. This is a topic for our future work.

TABLE VI
COMPUTATIONAL COMPLEXITY

		Algorithm	
		TAN	**HNB**
Training	**Time**	$O(tn^2 + k(nv)^2 + n^2 \log n)$	$O(tn^2 + k(nv)^2)$
	Space	$O(k(nv)^2)$	$O(k(nv)^2)$
Classification	**Time**	$O(kn)$	$O(kn^2)$
	Space	$O(knv^2)$	$O(k(nv)^2)$

REFERENCES

[1] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Commun. Surveys Tuts.*, vol. 10, no. 4, pp. 56-76, Fourth Quarter 2008.

[2] J. Erman, M. Arlitt, and A. Mahanti, "Traffic classification using clustering algorithms," in *MineNet 06: Proc. 2006 SIGCOMM workshop on Mining network data.* New York, NY, USA: ACM Press, 2006, pp. 281-286.

[3] J. Erman, A. Mahanti, M. Arlitt, I. Cohen, and C. Williamson, "Offline/realtime traffic classification using semi-supervised learning," *Perform. Eval.,* vol. 64, no. 9-12, pp. 1194-1213, October 2007.

[4] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, "Traffic classification on the fly," *ACM Special Interest Group on Data Communication (SIGCOMM) Computer Communication Review,* vol. 36, no. 2, 2006.

[5] A. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," in *ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS) 2005,* Banff, Alberta, Canada, June 2005.

[6] T. Auld, A. W. Moore, and S. F. Gull, "Bayesian neural networks for Internet traffic classification," *IEEE Trans. Neural Networks,* vol. 18, no. 1, January 2007, pp.223-239.

[7] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification," *Special Interest Group on Data Communication (SIGCOMM) Computer Communication Review*, vol. 36, no. 5, pp. 5-16, 2006.

[8] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "Blinc: Multilevel traffic classification in the dark," in *Proc. of the Special Interest Group on Data Communication conference (SIGCOMM) 2005,* Philadelphia, PA, USA, August 2005.

[9] J. Erman, A. Mahanti, and M. Arlitt, "Byte me: a case for byte accuracy in traffic classification," in *MineNet 07: Proc.3rd annual ACM workshop on Mining network data.* New York, NY, USA: ACM Press, June 2007, pp. 35-38.

[10] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in network identification of P2P traffic using application signatures," in *WWW2004,* New York, NY, USA, May 2004.

[11] L Jiang, H Zhang, and Z Cai, "A novel Bayes model: Hidden naive Bayes," *IEEE Trans. Knowl. Data Eng*, vol. 21, no. 10, pp. 1361-1371, October 2009.

[12] J. Zhang, Y. Xiang, Y. Wang, W. Zhou, Y. Xiang, and Y. Guan, "Network Traffic Classification Using Correlation Information," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no.1, pp. 104-117, January 2013.
[13] J. Zhang, C. Chen, Y. Xiang, W. L. Zhou and Y. Xiang. "Internet traffic classification by aggregating correlated nave bayes predictions," *IEEE Trans. Inf. Forensics Security*, vol. 8, no.1, pp. 5-15, 2013.
[14] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian Network Classifiers," *Machine Learning*, vol. 29, pp. 131-163, 1997.
[15] J Dougherty, R Kohavi, and M Sahami, "Supervised and unsupervised discretization of continuous features," *Machine Learning: Proceedings of the Twelfth International Conference*, 1995, Morgan Kaufmann Publishers, San Francisco, CA.
[16] R. C. Holte, "Very simple classification rules perform well on most commonly used datasets," *Machine Learning 11*, pp. 63-90.
[17] D. M. Chickering, "Learning Bayesian Networks is NP-Complete," *Learning from Data: Artificial Intelligence and Statistics V*, D. Fisher and H. Lenz, eds., pp. 121-130, Springer-Verlag, 1996.
[18] F. Zheng, and GI. Webb, "Semi-naive Bayesian Classification," *Journal of Machine Learning Research*, pp. 1-56, 2008.