Exploring Case-Based Bayesian Networks and Bayesian Multi-nets for Classification

Ahmed Hussein and Eugene Santos Jr.
Department of Computer Science and Engineering
University of Connecticut
Storrs, CT 06269-3155
ahussein@engr.uconn.edu and eugene@engr.uconn.edu

Abstract. Recent work in Bayesian classifiers has shown that a better and more flexible representation of domain knowledge results in better classification accuracy. In previous work [1], we have introduced a new type of Bayesian classifier called *Case-Based Bayesian Network (CBBN)* classifiers. We have shown that CBBNs can capture finer levels of semantics than possible in traditional Bayesian Networks (BNs). Consequently, our empirical comparisons showed that CBBN classifiers have considerably improved classification accuracy over traditional BN classifiers. The basic idea behind our CBBN classifiers is to intelligently partition the training data into semantically sound clusters. A local BN classifier can then be learned from each cluster separately. Bayesian Multi-net (BMN) classifiers also try to improve classification accuracy through a simple partitioning of the data by classes. In this paper, we compare our CBBN classifiers to BMN classifiers. Our experimental results show that CBBN classifiers considerably outperform BMN classifiers.

1 Introduction

Recent work in Bayesian classifiers has shown that better and more flexible representations of the relationships between domain attributes significantly improves classification accuracy. For example, in BN classifiers, the *Tree Augmented Naive-Bayes (TAN)* classifier relaxes the structure of the naive-Bayes classifier by approximating the interactions between the attributes using a tree-like structure. In addition, the *Bayesian network Augmented Naive-Bayes (BAN)* classifier extends TAN by allowing the attributes to form an arbitrary graph, rather than only a tree. Another type of BN classifier that permits even more flexible structures is the General Bayesian Network (GBN) classifier which treats the classification node as an ordinary node and identifies a relevant attribute subset around it determined by its Markov blanket. These classifiers have been shown to outperform naive-Bayes classifier [2, 3].

Bayesian Multi-net (BMN) classifiers are a generalized form of the augmented naive structure (i.e., TAN and BAN) in the sense that they allow different relationships among attributes for different values of the class variable. A BMN classifier consists of the prior probability distribution of the class variable C and a set of local networks, each corresponding to a value of C. Two forms of BMN classifiers have been examined. Tree Multi-net (BMN-TAN) [3], a generalization of TAN, where the structure of a local network is restricted to tree-like structure and Bayesian Multi-net (BMN-BAN) [4, 3], a generalization of BAN, where a local network can have unrestricted structure. Although the structures of these classifiers are strictly more expressive than traditional BN classifiers (i.e., TAN and BAN), experiments have shown that BMN classifiers perform as well as BN classifiers and that neither approach clearly dominates [3].

In previous work [1], we have introduced a new type of Bayesian classifier called *Case-Based Bayesian Network (CBBN)* classifiers. The basic idea behind CBBN classifiers is to intelligently partition the training data into semantically sound clusters. Each cluster is characterized and discriminated from other clusters by a unique assignment to its most relevant and descriptive attributes. This is called indexing. A local BN classifier can then be learned independently from each cluster conditioned on its index. Such an organization of the domain knowledge allows more precise and more relevant representation of the domain dependency relationships than possible in traditional BNs. We compared CBBN classifiers to BN classifiers with different structures (i.e., naive, TAN, BAN and GBN). Our empirical results showed that CBBN classifiers have considerably improved classification accuracy over BN classifiers.

In this paper, we further explore CBBN classifiers by comparing them to BMN classifiers. Our motivations behind this comparison are as follows: First, BMNs' partitioning of the data is simply restricted to the classes which may not fit the best (i.e., inherent) partitioning of the data and is only useful when the relationships among attributes are very different for different classes. In contrast, our CBBNs relax this restriction by permitting the use of *any appropriate* clustering methodology that discovers the best way to partition the data. Thereby, knowledge can be better represented and more accurate classifiers can be constructed. Second, the indices used by CBBNs provide an attribute selection procedure that is able to discard attributes that are irrelevant to classification. We show that CBBN classifiers outperform BMN classifiers. We also show that while BMN classifiers perform significantly worse than BN classifiers in some cases, our CBBN classifiers perform superior to or competitive with BN classifiers.

2 Our CBBN Approach - An Overview

We use a clustering technique to discover meaningful patterns represented by different clusters of data. Each cluster is then characterized and discriminated from other clusters by a unique assignment to its most relevant and descriptive attributes. This assignment is called an index. As we shall see, these indices provide a natural attribute selection for the CBBN classifiers. Intuitively, each cluster represents a piece of the domain knowledge described by the context of its index. These clusters can also be viewed as a set of conditionally independent cases with each case mapped to an index that describes the context of the knowledge relevant to that case. Because of the independence nature of the cases, the knowledge associated with each case can be represented separately by a BN classifier. This representation of the independent cases implies that the relationships among the corresponding attributes might be different for different cases. Thus, instead of assuming fixed relationships between attributes for the whole domain as in traditional BNs, these relationships can vary according to each different context of each case in the same domain. This conclusion is crucial, since it means that two variables X and Y might be directly dependent ($X \rightarrow Y$) in case C_i and independent in case C_j. Moreover, $X \rightarrow Y$ might occur in case C_i while $X \leftarrow Y$ occurs in case C_j. Even if the relationships in different cases are the same, the parameters that represent the strength of these relationships might be different.

Obviously, CBBNs subsume BMNs in that they exploit the inherent clusters in the data to partition the domain knowledge instead of relying on a simple/restricted partitioning of the data according to the classes which may not fit the natural data clusters and is only useful when the relationships among attributes is very different for different classes. Moreover, our CBBN approach provides a novel procedure for discarding irrelevant attributes. The attributes that constitute an index for a cluster have fixed values for all objects in the cluster. We conclude that these attributes are irrelevant to the classification task in this cluster. Therefore, a BN classifier learned from this cluster can safely exclude these attributes.

3 CBBN Classification Model

Constructing a CBBN classification model consists of the following three phases:

Clustering and Indexing: Let D be a data set described by a set of categorical attributes $A_1, A_2, ..., A_n, C$. A clustering algorithm is used to partition the training data set D into a set of clusters $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$ characterized by a set of mutually exclusive indices $\mathbf{I} = \{I_1, I_2, \ldots, I_k\}$ respectively. This indexing scheme guarantees at most one mapping per a data object to the set $\mathbf{I}$.

In order to generate such an indexing scheme, we begin by initializing $\mathbf{I}$ (a set of k n-dimension vectors) with “don't care” values for all elements of each vector I_i. For a particular cluster C_i, we compute the probability distribution for each attribute, i.e., the frequencies of its possible values estimated from the data in this cluster. We determine the value of each attribute that has the maximum frequency and assign this value to this attribute in I_i if its frequency exceeds an indexing threshold α. The resulting assignment is then used as a description of the objects in C_i, thus we move all objects that are not covered by I_i from C_i to the outliers cluster. The same procedure is repeated for each cluster. We then visit the outliers cluster to check for possible mappings of its objects back to the indexed clusters. These objects are retrieved from the outliers to be placed in a cluster if the objects are compatible to the cluster's description index.

In order to achieve mutual exclusion between the above assignments, we check each two assignments for the mutual exclusion condition (at least one common attribute is assigned differently). If they do not satisfy this condition, we search for the “don’t care” attribute in both assignments that can be assigned differently in both of them such that a minimum number of objects is rejected from both clusters due to the new assignments. We then update the members of all clusters, including the outliers, according to the new mutually exclusive assignments. Finally, to produce the index of each cluster, we simply discard any “don’t care” attributes in each cluster’s assignment. The algorithms for the above procedure can be found in [1].

Learning: We apply a BN learning algorithm to learn a BN classifier from the data objects in each indexed cluster. This local classifier, B_i where $i \in \{1, 2, ..., k\}$, is defined over a subset of attributes $V_i \subset \mathbf{V} = \{A_1, A_2, \ldots, A_n, C\}$. If $V(I_i)$ is the set of the attributes in I_i then $V_i = \mathbf{V} - V(I_i)$. We also learn a BN classifier, B_o, from the outliers cluster defined over the whole set $\mathbf{V}$.

Testing: We test the newly learned CBBN classification model on the given test data set T. Basically, we try to map each test object $(a_1, a_2, \ldots, a_n)$ in T to an index in $\mathbf{I}$ by comparing the attributes assignment in both of them. We then compute $P(C|a_1, a_2, \ldots, a_n)$ from the local BN classifier characterized by that index and assign to C the value that maximizes P. If an object cannot be mapped to any index in $\mathbf{I}$, we map it to B_o as the default classifier.

4 Experimental Results

4.1 Experiment Settings

In order to compare BMN classifiers to our CBBN classifiers, we have learned three types of BMN classifiers: BMN-TAN, BMN-BAN and BMN-BAN* based on three different learning algorithms: Chow-Liu [5], MDL score [6] and CBL2 [7] respectively [1]. We compare the classification accuracy of these classifiers to their corresponding CBBN classifiers (i.e., CBBN-TAN, CBBN-BAN and CBBN-BAN*).

These classifiers have been learned from each data set in a group of twenty-five benchmark data sets obtained from the UCI machine learning repository at (www.cs.uci.edu). In all data sets, objects with missing attribute values have been removed and numerical attributes have been categorized.

For data clustering in CBBNs, we used the clustering algorithm, *k-modes* [8]. This algorithm requires that the user specify the number of clusters k. In this work, we have determined an acceptable range of k for each data set. More specifically, k can take integer values between $k_{min} = 2$ and k_{max} which is the maximum number of clusters estimated such that each cluster has a number of objects sufficient to learn a BN classifier. We then ran our experiments at three different values of k (k_{min}=2, k_{max}, and $k_{arb} \in]k_{min}, k_{max}[$). For the indexing threshold α, we consider an attribute as a descriptive attribute if its maximum frequency is not less than $\alpha_{min} = 0.7$. We then arbitrarily select α for each data set within the range $[\alpha_{min}, 1]$.

[1] naive and GBN structures are not defined for BMN approach.

4.2 Classification Accuarcy

Due to space limitations we only show results at $k = k_{arb}$. Similar results have been obtained at $k = k_{min}$ and at $k = k_{max}$. Our experimental results are presented in Table 1. This table shows the classification accuracy for three different classifier's structures (TAN, BAN and BAN*) built based on three different approaches (BN, BMN and CBBN). In most data sets (21 data sets), BMN classifiers give classification accuracy similar to their corresponding structures in BN classifiers. However, in *letter* data set, BMN classifiers work better than BN classifiers while in the three data sets left (e.g. *soybean-large*) they work considerably worse. A quick inspection of the networks learned in *letter* and *soybean-large* reveals that the dependency assertions among the attributes are very different for different classes in *letter* while almost the same in *soybean-large*. In contrast, CBBN classifiers almost always show considerable improvement over BN classifiers or show similar performance (i.e., classification accuracy). This suggests that while simple partitioning of the data by classes used in BMN classifiers is not effective in many cases, our semantically sound clustering almost always leads to performance improvement. Table 1 also shows that our CBBN classifiers perform considerably better than non-bayesian ML classifiers such as C4.5 and IB classifiers.

Table 2 confirms these results by comparing BMN classifiers to BN classifiers. The results in this table reveals that BMN classifiers do not show an average improvement in accuracy over BN classifiers and do not frequently beat them. Furthermore, we have conducted two comparisons between CBBN classifiers and BMN classifiers. In the first comparison, Table 3, we measure the average improvement in accuracy of CBBN classifiers over their corresponding BMN classifiers. This table shows that CBBN classifiers considerably outperform BMN classifiers for all structures. In the second comparison, Table 4, we measure the average improvement in accuracy of CBBN-BAN* (the best classifier we have in our CBBN model) over the different structures of BMN classifier. This comparison shows that CBBN-BAN* classifier has a significant average improvement in accuracy over BMN classifier for all structures.

4.3 Time Cost

Table 5 shows the construction time of CBBN and BMN classifiers for five selected data sets. The results show that CBBN is more computationally expensive than BMN. This is due to the time needed to accomplish the intelligent partitioning (clustering and indexing). The time cost for the k-modes clustering algorithm is $O(tnkN)$ where t is the number of iterations for k-modes to converge and N is the size of the training set. Obviously, the clustering time is linear in N and k. However, the indexing process is time-consuming, but it may reduce the time cost of the learning process from each cluster because of the smaller number of attributes involved in learning compared to BMN models. A comparison between the time cost of CBBN and BMN models will depend on the number of clusters k and the number of classes c from which we build the local networks.

Table 1: % Classification accuracy for ML, BN, BMN and CBBN classifiers

	Datasets							**ML**		**TAN**			**BAN**			**BAN***		
no.	name	n	c	train	test	k_{arb}	α	C4.5	IB	BN	BMN	CBBN	BN	BMN	CBBN	BN	BMN	CBBN
1	australian	14	2	690	CV-5	3	0.80	85.217	81.739	81.159	81.884	87.391	86.957	86.667	96.232	87.246	87.681	**97.101**
2	breast	10	2	683	CV-5	3	0.80	94.436	96.047	95.900	95.900	95.608	96.633	96.340	**98.682**	96.779	96.486	**98.682**
3	car	6	4	1728	CV-5	4	0.85	69.329	66.204	94.097	94.329	**97.280**	90.451	90.394	96.586	94.039	95.081	96.933
4	chess	36	2	2130	1066	4	0.78	**99.390**	95.028	92.495	92.401	95.872	94.090	96.435	96.998	94.184	93.809	96.717
5	cleve	13	2	296	CV-5	2	0.90	73.986	77.027	79.730	79.392	93.581	79.392	80.068	93.243	82.095	81.757	**95.608**
6	crx	15	2	653	CV-5	3	0.95	86.217	77.489	83.920	84.074	94.334	86.524	86.371	94.793	88.055	89.433	**95.100**
7	diabetes	8	2	768	CV-5	3	0.75	76.172	71.484	75.000	74.349	87.891	75.520	75.911	87.500	77.083	78.776	**92.188**
8	DNA	60	3	2000	1186	4	0.70	92.580	75.801	93.592	94.772	96.374	90.135	91.737	**97.218**	88.533	88.533	96.121
9	flare	10	2	1066	CV-5	3	0.80	82.551	82.833	82.552	81.989	89.587	82.645	82.739	94.090	82.833	81.426	**94.934**
10	german	20	2	1000	CV-5	3	0.82	72.300	69.700	72.200	71.800	91.500	73.200	72.400	92.400	76.800	77.100	**94.800**
11	glass	9	7	214	CV-5	2	0.80	62.241	70.561	68.961	69.159	85.514	70.561	65.421	95.794	71.028	68.692	**96.262**
12	heart	13	2	270	CV-5	2	0.87	80.471	80.000	83.704	83.333	92.593	82.963	83.704	94.815	86.296	87.407	**95.185**
13	led24	24	10	200	3000	2	0.85	65.567	39.433	73.800	73.800	82.967	72.600	70.600	**95.767**	74.100	75.200	94.600
14	liver	6	2	345	CV-5	3	0.88	60.870	64.348	65.217	66.377	79.420	66.957	69.275	94.493	67.246	68.114	**95.072**
15	letter	16	26	15000	5000	10	0.85	77.700	72.800	83.460	84.380	94.920	76.640	80.120	91.440	79.300	81.200	**96.060**
16	mofn-3-7-10	10	2	300	1024	3	0.90	85.449	89.355	91.797	91.602	94.727	86.328	86.523	95.508	88.477	88.379	**96.680**
17	nursery	8	2	8640	4320	6	0.80	68.241	66.157	91.713	92.292	95.255	91.296	90.787	**97.593**	93.079	92.685	97.199
18	pima	8	2	768	CV-5	3	0.75	75.130	68.750	74.870	75.521	86.328	74.740	76.432	92.318	79.297	79.036	**93.359**
19	satimage	36	6	4435	2000	5	0.85	83.100	88.800	77.600	78.000	92.500	80.550	77.150	95.750	84.450	80.000	**96.050**
20	segment	19	7	1540	770	3	0.90	93.506	96.104	85.455	82.857	94.805	91.039	90.260	95.584	90.390	91.169	**96.104**
21	shuttle-small	9	7	3866	1934	5	0.77	99.121	**99.586**	98.914	98.862	97.156	98.914	98.966	98.190	97.208	96.329	96.794
22	soybean-large	35	19	562	CV-5	3	0.85	91.993	90.747	58.363	53.559	71.174	92.349	87.189	**96.263**	92.865	89.502	95.196
23	vehicle	18	4	846	CV-5	3	0.85	69.740	63.830	67.967	66.548	74.470	67.494	64.303	**93.498**	71.631	66.785	87.589
24	vote	16	2	435	CV-5	3	0.80	95.172	94.713	88.966	89.655	94.943	90.115	90.345	94.253	95.632	94.943	**97.241**
25	waveform21	21	3	300	4700	2	0.90	74.787	75.766	75.383	73.830	92.553	77.723	76.809	94.553	78.787	78.191	**94.574**
	best result count							1	1	0	0	1	0	0	5	0	0	**16**

Table 2: BMN vs. BN

BMN→	**TAN**	**BAN**	**BAN***
% win/BN	44	48	40
% imp/BN	-0.409	-0.467	-0.477

Table 3: CBBN vs. BMN

CBBN→	**TAN**	**BAN**	**BAN***
% win/BMN	92	96	100
% imp/BMN	12.303	16.268	14.337

Table 4: CBBN-BAN* vs. BMN

CBBN-BAN*	**TAN**	**BAN**	**BAN***
% win/BMN	96	96	100
% imp/BMN	19.672	16.851	14.337

Table 5: Construction time in CPU seconds

	data set					**TAN**		**BAN***	
no	name	n	N	c	k_{arb}	BMN	CBBN	BMN	CBBN
1	chess	36	2310	2	4	56	73	66	87
2	DNA	60	2000	3	4	212	374	608	957
3	nursery	8	8640	2	6	20	26	22	33
4	flare	10	1066	2	3	5	10	11	13
5	vote	16	435	2	3	6	14	16	18

5 Conclusions and Futurework

We have compared our CBBN classifiers to BMN classifiers. We have shown that our CBBN classifiers have considerably better classification accuracy than BMN classifiers.

We plan to extend this work in the following direction: We suggested using the k-modes clustering algorithm and ran our experiments with three different values of k and with α arbitrarily selected by the user. Although we obtained good results in all runs, there is no guarantee that these are the best possible results. We would like to come up with a procedure to optimize k and α for the best classification accuracy.

6 Acknowledgements

This work was supported in part by Air Force Office of Scientific Research Grant No. F49620-03-1-0014.

References

1. Santos, E., Hussein, A.: Case-based bayesian network classifiers. In: Proceedings of the Seventeenth International FLAIRS Conference, AAAI Press (2004)
2. Cheng, J., Greiner, R.: Comparing bayesian network classifiers. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence. (1999)
3. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning **29** (1997) 131–161
4. Cheng, J., Greiner, R.: Learning bayesian belief network classifiers: Algorithms and system. In: Proceedings of the Fourteenth Canadian Conference on Artificial Intelligence. (2001)
5. Chow, C.K., Liu, C.: Approximating discrete probability distributions with dependence trees. IEEE Transaction on Information Theory **14** (1968) 462–467
6. Lam, W., Bacchus, F.: Learning bayesian belief networks: An approach based on the mdl principle. Computational Intelligence **10** (1994)
7. Cheng, J., Bell, D., Liu, W.: Learning bayesian networks from data: An efficient approach based on information theory. In: Proceedings of the Sixth ACM International Conference on Information and Knowledge Managment. (1997)
8. Huang, Z.: Extensions to the k-means algorithm for clustering large data sets. Data Mining and Knowledge Discovery **2** (1998) 283–304