Learning Dynamic Naive Bayesian Classifiers*

Miriam Martínez
Instituto Tecnológico de Acapulco
Acapulco, Guerrero, Mexico
miriamma_ds@hotmail.com

L. Enrique Sucar
Instituto Nacional de Astrofísica, Óptica y Electrónica
Tonantzintla, Puebla, Mexico
esucar@inaoep.mx

Abstract

Hidden Markov models are a powerful technique to model and classify temporal sequences, such as in speech and gesture recognition. However, defining these models is still an art: the designer has to establish by trial and error the number of hidden states, the relevant observations, etc. We propose an extension of hidden Markov models, called *dynamic naive Bayesian classifiers*, and a methodology to learn automatically these models from data. The method determines: (i) the number of hidden states, (ii) the relevant attributes, (iii) the best discretization, and (iv) the structure of the model. Experimental results on learning different dynamic naive Bayesian classifiers for gesture recognition, show that our method improves significantly the recognition rates, and at the same time obtains simpler models.

Introduction

Hidden Markov models (HMMs) (Rabiner & Juang 1993) have become the standard method for modeling and recognizing temporal sequences under uncertainty, such as in speech recognition and gesture classification. However, defining these models is still essentially an art. The designer has to establish the number of states, the relevant observation attributes, and, in some cases, the discretization of continuous observations. Recently, in the context of gesture recognition (Aviles, Sucar, & Mendoza 2006), we proposed an extension of HMMs called *dynamic naive Bayesian classifiers* (DNBC). A DNBC decomposes the observation node into a set of attributes considered independent given the class. This simplifies the model when there are many observation values, and allows selecting only the relevant attributes. In this paper we develop a methodology to learn automatically a DNBC from data. The method determines: (i) the number of hidden states, (ii) the relevant attributes, (iii) the best discretization, and (iv) the structure of the model. This methodology helps the designer to automatically obtain the best models, which in general are better (higher classification rates) and simpler (fewer attributes) than those obtained just by trial and error.

We have tested this method in the recognition of seven manipulative gestures (Montero & Sucar 2006): (*Opening a drawer, Erasing, Writing, Reading a book, Using a mouse, Turning on printer and Using a telephone*). For each gesture, our method produces a very simple and efficient classifier with 98% accuracy in average, a significant improvement over the original HMMs.

Related Work

We briefly review related work in disctretization and structure learning.

Discretization

Discretization methods for classifiers can be divided into two main types: (i) unsupervised, and (ii) supervised. Unsupervised methods do not consider the class variable, so the continuous attributes are discretized independently. Supervised methods consider the class variable, so that the division points are selected in function of the value of the class for each data point. The problem of finding the optimal number of intervals and the corresponding limits can be seen as a search problem. That is, we can generate all possible division points over the range of each attribute (where there is a change of class), and estimate the classification error for each possible partition. Unfortunately, generating and testing all possible partitions is impractical. In the worst case, there are in the order of 2^{MN} possible partitions, where M is the number of attributes and N is the number of possible partition points per attribute.

For Bayesian classifiers, (Pazzani 1995) introduces a method that, starting from an initial partition, it makes an iterative search for a better one, by joining or splitting intervals, and testing the classification accuracy after each operation. (Friedman & Goldszmidt 1996) do discretization while learning the structure of a Bayesian network. For a given structure, a local search procedure finds the discretization for a variable that minimizes the description length in relation to the adjacent nodes in the graph, and this is repeated iteratively for each continuous variable. Valdes et al. (Valdes, Molina, & Peris 2003) present a technique based on evolution strategies. The attributes are discretized to maximize class predictability. This method leads to simple models and can discover irrelevant attributes. A survey of dis-

*This work is supported in part by CONACYT Project No. 47968.

cretization methods for naive Bayes classifiers is given in (Yang & Webb 2002).

Most methods consider a classifier in which the class is known, while in our case the state variable is a hidden node.

Structure learning

Our method is related, on one hand, to work on improving naive Bayes classifiers; on the other hand, on general techniques for learning dynamic Bayesian networks.

The naive Bayes classifier assumes that the attributes are independent given the class. If this is not true, there are two basic alternatives. One is to transform the structure of the classifier to a Bayesian network, by introducing directed arcs between the dependent attributes (Friedman, Geiger, & Goldszmidt 1997). The disadvantage is that the simplicity of the NBC is lost. The other alternative is to transform the structure maintaining a star or tree-structured network. For this, (Sucar 1992) introduces 3 basic operations: (i) eliminate an attribute, (ii) join two attributes into a new combined variable, (iii) introduce a new attribute that makes two dependent attributes independent (hidden node). These operations are based on statistical tests to measure the correlation of pairs of attributes given the class variable. In a more recent work, Martínez (Martínez & Sucar 2006) extends the method incorporating discretization to optimize a naive Bayes classifier. Pazzani (Pazzani 1996) proposes an alternative algorithm for variable elimination and merging. The algorithm is based on two search procedures: (i) forward sequential selection and joining and (ii) backward sequential elimination and joining. It starts from a full (empty) structure, and selects attributes for elimination (addition) or for combination, testing the classification accuracy after each operation. The advantage of these approaches is that they preserve the simplicity and efficiency of the NBC. Previous work considers a static NBC, in which the class variable is observable; here we extend these approaches to a dynamic model with a hidden state.

There are several approaches for learning dynamic Bayesian networks (DBN) (Friedman, Murphy, & Russell 1998; Campos & Puerta 2000), which, in general, first learn the base structure and then the transition model. Although this work is somewhat related to the general problem of learning DBN, there are important differences: (i) a particular type of structure is considered, so in this sense the search space is reduced, (ii) general learning algorithms for DBN do not consider eliminating or joining variables.

Dynamic Naive Bayesian Classifiers

A dynamic naive Bayesian classifier can be seen as an extension of HMMs or a particular case of a dynamic Bayesian network. It is like a HMM in which the observation node has been decomposed in a number of attributes that are consider independent given the state. From a DBN perspective, at each time there is a naive Bayes classifier (base structure), and the state variables are connected from one time to the next (transition structure). An example of a DNBC is depicted in figure 1.

From the perspective of modeling complex temporal sequences, a DNBC has two main advantages over a HMM:

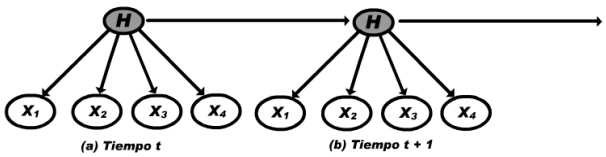

Figure 1: A dynamic naive Bayesian classifier. Two time intervals are shown (t, $t+1$), with a base classifier with 4 attributes ($x_1 - x_4$).

(i) when there is a large number of observation symbols, it allows for a considerable reduction in the number of parameters, and consequently a reduction in the number of examples required to train the model; (ii) it opens the door for selecting the relevant attributes, as we will show later. The possible drawback could be that the attributes are not really conditionally independent given the state, however this problems is reduced by the structure improvement phase in our learning methodology.

The algorithms for inference (backward/forward) and parameter learning (Baum-Welch) for DNBCs are straightforward extensions of the ones for HMMs (Rabiner & Juang 1993). The novel aspect is learning the structure, as we will see in the following section.

Learning the Model

As for dynamic Bayesian network, learning a DNBC can be divided into two parts:

1. learning the base structure,
2. learning the transition model.

Our work focuses on learning the base structure. Once this is defined, the transition structure consists just of a directed link from the state at one time to the next, $S_t \rightarrow S_{t+1}$; so the emphasis in this work is on learning the base structure and not the transition model.

The methodology to learn a DNBC is summarized in Algorithm 1. It consists of 5 phases: (i) initialization, (ii) discretization, (iii) hidden states estimation, (iv) structural improvement, (v) dynamic model integration. Following each phase is described.

Initialization

This step is done once to build the initial base structure. It considers all the attributes and an initial partition for the continuous attributes with two equal width intervals. The initial number of hidden states is set to two. The parameters for this initial structure are estimated from the training data using the Expectation–Maximization algorithm (EM) (Dempster, Laird, & Rubin 1977).

Discretization

Given the current structure, in this stage the discretization for each continuous attribute is optimized. From the initial discretization, it generates additional partitions based on the MDL principle. Each attribute is processed independently, by splitting each interval into two parts, and testing each

Algorithm 1 Learning a dynamic Bayesian classifier: high–level description.

Require: $C(B,T)$: DNBC, B: base estructure, T: transition structure, N; number of hidden states, A: attribute configuration, D_i: discretization of attribute i
Initialization: $B_1(N,A,D)$, $N=2$, A all the attributes, D_i=2
Obtain parameters for B_1 using EM
repeat
Discretization: improve discretization, D, based on MDL
Hidden states estimation: estimate number of hidden nodes, N, based on MDL
Structural improvement: improve base structure, B, via variable elimination and combination
until base structure can not be improved
Dynamic model integration: obtain T
Ensure: Return final model: $C_F(B,T)$

partition using MDL. From these, the split with the best measure is selected; and the process is repeated with the new partition iteratively until the MDL can not be improved. The MDL measure makes a compromise between accuracy and complexity. The measure we use is similar to the one proposed in (Lam & Bacchus 1994), which estimates the accuracy ($NetWeight$) by measuring the mutual information between the attribute and the class; and the complexity ($NetLength$) by counting the number of parameters required. A constant α, in [0, 1], is used to balance the weight of each aspect, accuracy vs. complexity. Thus, the *quality* measure is:

$$Quality = (1-\alpha) * \left(1 - \frac{NetLength}{MaxLength}\right) + \alpha * \left(\frac{NetWeight}{MaxWeight}\right) \quad (1)$$

Where $NetLength$ is proportional to the number of parameters required in the model, and $NetWeight$ corresponds to the sum of the mutual information between each attribute and the class; which gives an estimate of the model precision. The maximum length, $MaxLength$, and weight, $MaxWeight$, are estimated by considering the maximum number of intervals per attribute. An $\alpha = 0.5$ gives equal weight to accuracy and complexity, while an α close to 1 gives more importance to accuracy.

Hidden states estimation

In this stage, the number of values for the hidden class node is obtained. From the initial structure, it generates additional partitions in the class node and testing each structure using again the MDL measure. The process is repeated with a new partition iteratively until the MDL can not be improved. The MDL measure is the same one used in the discretization phase.

Structural improvement

Given the current discretization and number of hidden states, in this phase the base structure is improved to eliminate superfluous attributes and eliminate or combine dependent attributes.

To alter the structure we consider two operations:

- Attribute elimination: one of the attributes is not considered in the base structure.
- Attribute combination: two attributes are combined into a new attribute whose values are the cross product of the original attributes.

These operations preserve the simple star–like structure of the naive Bayes classifier; performing feature selection and, at the same time, reducing the potential impact of non–independent attributes. This phase is described in algorithm 2.

Algorithm 2 Structural improvement phase.

Require: B: current base structure, N; number of hidden states, A: attribute configuration, D: current discretization, T_1, T_2: thresholds
Obtain mutual information between each attribute and the state variable, $I(A_i,S)$
Obtain the conditional mutual information between each pair of attributes given the state, $CI(A_i,A_j)$
repeat
Eliminate irrelevant attributes: eliminate A_i if $I(A_i,S) < T_1$
Eliminate/combine dependent attributes:
if $CI(A_i,A_j) > T_2$ **then**
Eliminate A_i, obtain B_1
Eliminate A_j, obtain B_2
Join A_i, A_j, obtain B_3
Select the structure, B_j with lower MDL
end if
until no attributes can be eliminated or combined
Ensure: Return "best" structure: B_F

These 3 phases, discretization–hidden state estimation–structural improvement, are repeated iteratively until convergence (no significant changes in the model). These 3 phases are interrelated and together produce a huge combinatorial space. The order of this operations can have an impact on the results. Our solution is basically a heuristic search in this space, which has demonstrated good results in practice.

Dynamic Model Integration

Once the base structure is obtained in the previous stages, the transition model is set as a directed link between consecutive states, and the parameters of the complete model are obtained using a direct extension of the Baum–Welch algorithm (Rabiner & Juang 1993).

Note that the model cannot be evaluated (with test data) until the end of this phase, so previous stages are based on indirect quality measures, mainly MDL.

Experimental Results

Experimental setup

The method was evaluated in the visual recognition of 7 manipulative gestures. These hand gestures consist of hand

movements done by a person while interacting with different objects in an office environment. The gestures considered are the following: *Opening a drawer, Erasing, Writing, Reading a Book, Using a Mouse, Turning on a printer and Using a Telephone*. The gestures were captured using a ceiling mounted camera, as shown in figure 2. A visual processing system (Montero & Sucar 2006) tracks the (right) hand of the person, and obtains the following 5 attributes per image: *X* coordinate and *Y* coordinate of the centroid of the hand region, and the changes in magnitude, direction and velocity. All five attribute are continuous.

Figure 2: Examples of the gestures *opening the drawer* (left) and *using the mouse* (right). For each gesture the trajectory followed by the hand centroid is shown.

100 examples (videos) of each gestures were captured, 50 sequences for training and 50 for testing, per gesture. These videos were recorded in a laboratory environment with changing lighting conditions. The system was implemented in a PC with a Pentium IV processor at 3.1 GHz, 1 GB RAM, programmed in Java, with some classes from the Bayesian networks toolkit ELVIRA (Consortium 2002).

Results

A model for each gesture was learned based on the proposed methodology. For the MDL quality measure we used an $\alpha = 0.5$. In these experiments, one iteration of the algorithm was enough to obtain a very good performance (recognition rates), so other iterations were not necessary. Each model is evaluated in terms of recognition rates on the test data, and compared to a HMM with 2 hidden states, trained and tested on the same data sets.

First we illustrate the method, in particular the structural improvement phase, on one gesture: *Opening a drawer*. In the case of the model for the gesture *Opening a drawer*, after discretization and hidden state estimation (7 states), the operations for structural improvement are shown in table 1. Initially the *velocity* attribute is eliminated, then the *Y* coordinate, and finally the *X* coordinate. The final model considers only two attributes: *Magnitude* and *Direction*. The resulting structure is similar for the other gestures, although the discretization, number of states and attributes vary.

Table 2 shows the recognition rates for the final DNBC model for each gesture, as well as the attributes included in the resulting structure. It also presents the average recognition rate. For comparison, table 3 depicts the individual and average recognition rates obtained with HMMs for each gesture.

	Operation	Attributes
0		X, Y, Magnitude, Direction, Velocity
1	Elim. Vel.	X, Y, Magnitude, Direction
2	Elim. Y	X, Magnitude, Direction
3	Elim. X	Magnitude, Direction

Table 1: Structural improvement for learning the gesture *Opening drawer*. For each stage, the operation performed and the resulting attributes are shown.

Gesture	Attributes	Precision in %
Opening a drawer	Mag, Dir	100%
Erasing	Dir, Vel	94%
Writing	Dir, Vel	98%
Reading a book	Mag, Dir	98%
Turning on a printer	Mag, Dir	100%
Using a mouse	Dir, Vel	98%
Using a telephone	Mag, Dir	100%
Average		98.25%

Table 2: Final DNBC models obtained. The attributes and recognition rates are shown for each gesture, and also the average recognition.

Gesture	Precision in %
Opening drawer	100%
Erasing	80%
Writing	100%
Reading a book	100%
Turning on a printer	96%
Using a mouse	78%
Using a telephone	100%
Average	93.48%

Table 3: Recognition rates for HMMs per gesture, and average. Each HMM considers the 5 attributes.

We observe a significant average improvement in the recognition rates of 5 points, or nearly 10%, with respect to the HMMs. Also, the resulting models are simpler, as they only consider two attributes, although these are not the same for the different gestures. An important observation is that the best models for each gesture vary in terms of discretization, number of states and attributes; so it will be very difficult to obtain these models if they are specified by the designer.

Conclusions and Future Work

In this paper we develop a methodology to learn automatically a dynamic naive Bayesian classifier from data. The method determines: (i) the number of hidden states, (ii) the relevant attributes, (iii) the best discretization, and (iv) the structure of the model. This methodology has been evaluated in modeling and recognizing 7 hand gestures, based on visual attributes. The resulting models have shown an improvement in recognition rates compared to HMMs, and at the same time are simpler. Thus, our method provides an au-

tomatic way to design models for dynamic processes, such as in speech and gesture recognition, freeing the designer from a long and tedious trail and error process. The learning methodology can also be applied for HMMs, by just taking out the structural improvement phase

We have recently developed an alternative approach to learn DNBC based on evolutionary computation (see paper by Palacios-Alonso et al. in this volume), and expect to apply this approach in other domains, in particular for robotics.

Acknowledgements

We thank Antonio Montero for providing the data on gesture recognition and for his help in the experimental evaluation. We also thank the anonymous reviewers for their comments that helped to improve the paper.

References

Aviles, H.; Sucar, L.; and Mendoza, C. 2006. Visual recognition of similar gestures. *Proceedings of the International Conference on Pattern Recognition* 1100–1103.

Campos, L., and Puerta, J. 2000. Learning dynamic belief networks using conditional independence tests. *International Conference on Infortmation Processing and Management of Uncertainty* 325–332.

Consortium, E. 2002. Elvira: an environment for creating and using probabilistic graphical models. In *Proceedings of the First European Workshop on Probabilistic Graphical Models (PGM'02)*.

Dempster, A.; Laird, N.; and Rubin, D. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B* 39:1–38.

Friedman, N., and Goldszmidt, M. 1996. Discretizing continuous attributes while learning bayesian networks. *13th International Conference on Machine Learning (ICML)* 157–165.

Friedman, N.; Geiger, D.; and Goldszmidt, M. 1997. Bayesian network classifiers. *Machine Learning* 29:131–163.

Friedman, N.; Murphy, K.; and Russell, S. 1998. Learning the structure of dynamic probabilistic networks. In *Proc. Fourteenth Conference on Uncertainty in Artificial (UAI'98)*, 139–147.

Lam, W., and Bacchus, F. 1994. Learning bayesian belief netwoks: An aproach based on the mdl principle. *Computational Intelligence* 10:269–293.

Martínez, M., and Sucar, L. 2006. Learning an optimal naive bayes classifier. *Proceedings of the International Conference on Pattern Recognition* 1236–1239.

Montero, J. A., and Sucar, L. E. 2006. Context-based gesture recognition. *Lecture Notes in Computer Science* 4225:764–773.

Pazzani, M. J. 1995. An iterative improvement approach for the discretization of numeric attributes in bayesian classifiers. *KDD95* 228–233.

Pazzani, M. J. 1996. Searching for atribute dependencies in bayesian classifiers. *In preliminary Papers of the Intelligence and Statistics* 424–429.

Rabiner, L., and Juang, B. 1993. Fundamentals of speech recognition. *Signal Processing Series, Prentice Hall, New Jersey, USA.*

Sucar, L. 1992. *Probabilistic reasoning in knowledge based vision systems.* PhD dissertation, Imperial College of Science, Technology, and Medicine, London U.K.

Valdes, J.; Molina, L.; and Peris, N. 2003. An evolution strategies approach to the simultaneous discretization of numeric attributes in data mining. In *Proc. of the World Congress on Evolutionary Computation*, 1957–1964. Canberra, Australia: IEEE Press.

Yang, Y., and Webb, G. 2002. A comparative study of discretization methods for naive-bayes classifiers. In *Proc. of the Pacific Rim Knowledge Acquisition Workshop (PRKAW)*.