

A Dynamic Bayesian Network Based Structural Learning towards Automated Handwritten Digit Recognition

Olivier Pauplin and Jianmin Jiang

Digital Media & Systems Research Institute
University of Bradford, United Kingdom
o.pauplin@bradford.ac.uk, j.jiang1@bradford.ac.uk
<http://dmsri.inf.brad.ac.uk/>

Abstract. Pattern recognition using Dynamic Bayesian Networks (DBNs) is currently a growing area of study. In this paper, we present DBN models trained for classification of handwritten digit characters. The structure of these models is partly inferred from the training data of each class of digit before performing parameter learning. Classification results are presented for the four described models.

1 Introduction

Bayesian Networks (BNs) [1][2][3], also called Belief Networks or Probabilistic Networks, allow to represent probability models in an efficient and intuitive way. Their temporal extension, Dynamic Bayesian Networks (DBNs) [4][5], have been recently applied to a range of different domains. A natural way to use them is to represent temporal cause-and-effect relationships between events. In that case, the DBN can be seen as a knowledge base for diagnostic and prediction [6]. Another kind of application exploits the ability of DBNs to be trained to detect patterns. They have been used in speech recognition [7] as a flexible and efficient extension of Hidden Markov Models. In video or image analysis, DBNs can be used to detect specific actions, with applications such as sport event detection [8], meeting activity segmentation [9] (also based on prosodic and lexical features), and surveillance and tracking [10]. For most of these tasks, the observed information used as an input for the DBN is made of pre-extracted features (various colour statistics, motion intensity of tracked points...), however it is also possible to use low-level data such as image pixels, as shown for instance by the application of DBNs to character recognition [11].

In this paper, we perform handwritten digit recognition using DBN models whose structure have been partly automatically learned from the training set of data. Complete structure learning is tractable for BNs or DBNs with complete data [12][13][14], but becomes rapidly intractable for DBNs with numerous nodes and with the presence of hidden nodes. To overcome this problem, we adopted a mixed approach where some links of the graph are learned independently from the final structure of the model, and are then completed with additional nodes

and links. The paper is organised as follows: in section 2, we briefly introduce BNs and DBNs, their main properties and the main ideas behind parameter learning in DBNs. We then describe the four kinds of DBN models we built to conduct the experiments. Section 3 contains the description of the experiments, and the results. Section 4 concludes this paper.

2 Dynamic Bayesian Networks

2.1 General Remarks

A Bayesian Network is a Directed Acyclic Graph (DAG) in which each node represents a random variable and quantitative probability information. Directed links (arrows) connect pairs of nodes. Node X is said to be the parent of node Y if there is an arrow from X to Y . The meaning of that link can be expressed in the following way: “ X has a direct influence on Y ”. Influence is quantified by Conditional Probability Tables (CPT, for discrete nodes) or Conditional Probability Densities (CPD, for continuous nodes) associated with each node, which contain the conditional probabilities of the random variable for each combination of values of its parents. Nodes that do not have parents are associated with a prior probability instead of a conditional probability.

Given a set of random variables (X_1, \dots, X_n) , the value of every entry in the full joint probability distribution is given by equation 1. A key aspect of BNs is that the equation can be written in a simplified form which is easily deduced from the topology of the network. The joint probability distribution in a BN is given by equation 2, where $Parents(X_i)$ denotes the parents of X_i .

$$P(X_1, \dots, X_n) = P(X_1) \cdot \prod_{i=2}^n P(X_i | X_{i-1}, \dots, X_1) \quad (1)$$

$$P(X_1, \dots, X_n) = \prod_{i=1}^n P(X_i | Parents(X_i)) \quad (2)$$

Dynamic Bayesian Networks are an extension of static BNs to temporal events occurring at discrete times t . The graph of a DBN is built by juxtaposing several instances of a single BN, each instance of the BN (or time slice of the DBN) corresponding to a value of t . The complete network is obtained by adding directed links between slices (inter-slice links), in the direction of the time flow. Two assumptions are widely used in order to simplify the modelling:

- The Markov order of the DBN is equal to 1: the states at t only depend on states at t and $t - 1$;
- The DBN is stationary: its structure and parameters (conditional and prior probabilities) are the same for all $t \geq 2$ (provided the first time slice is $t = 1$).

DBNs can be used to perform inference, training and pattern detection tasks on data sets. Inference is the computation of the posterior probability of a query

variable in the network, given the values of observed variables. Inference can be exact or approximate [4], approximate inference being the only feasible solution for large networks. One advantage of using DBNs over BNs is that the stationarity assumption allows to apply efficient inference algorithms (an unrolled DBN treated as a static BN would require much more computational power).

A detailed description of parameter and structure learning is beyond the scope of this article (see [15][16][17] for more details). For a discrete node related to a single random variable, learning the parameters means finding the conditional probabilities of the random variable for each different combination of values of its parents. For a continuous Gaussian node, the same principle applies except that two parameters have to be determined in each case: the mean and the standard deviation of the Gaussian distribution. Learning the parameters with complete data can be done by applying the principle of Maximum Likelihood, i.e. trying to find the set of parameters $\{\theta_1, \dots, \theta_m\}$ that maximises the likelihood $P(d_1, \dots, d_p | \theta_1, \dots, \theta_m)$ where $\{d_1, \dots, d_p\}$ is a set of observed data. When some variables are hidden (or data is incomplete), the Expectation-Maximisation (EM) algorithm can be used to learn the DBN parameters: using inference as a subroutine, it computes the expected values of the hidden variables given the current parameter values and the data, then it uses the inferred values as if they were observed values and applies the principle of Maximum Likelihood. Those steps are iterated until no improvement of the likelihood is achieved.

2.2 The Proposed Model Designs

The DBN models we present in this study have their structure partly based on inter-slice links learned from data. We have built and tested the following four models.

Model 1: Learned inter-slice links. The task of learning a DBN structure relies on the choice of a scoring function assessing how well a network “matches” a set of data. That scoring task can be performed by using the Bayesian Information Criterion (BIC score) [17], which combines the likelihood of the data according to a network with a penalty intended to avoid learning a network with too high a complexity (the absence of penalty would lead in most cases to learning the completely connected network). Learning the structure of DBNs from incomplete data may be done using the Structural EM algorithm (SEM) [18], but remains computationally challenging for complex networks.

The BIC score for a graph G and a set of complete data D is given by:

$$BIC = \log P(D|G, \hat{\Theta}) - \frac{\log N_s}{2} N_p \quad (3)$$

where:

- $\hat{\Theta}$ is the set of parameters for G that maximises the likelihood of D ,
- N_s is the number of data sample in D ,
- N_p is the dimension of G or number of free parameters, which is equal to the number of parameters in the case of complete data.

Based on this approach, Model 1 is made of a structure whose only links are inter-slice links between evidence nodes, learned from the complete data. Structure learning is performed separately for each class of character, using a random subset of 500 digits per class extracted from our training set (the MNIST database, presented in section 3). The evidence nodes (or observed nodes) of the DBN observe values from a column of pixels of a digit image, i.e. node E_i^t observes the value of the pixel at the intersection of line i and column t . That approach may lead to DBNs with a number of nodes such that the task of inference and learning would be difficult. In order to reduce the number of parameters of our models, we normalised the digit images to the size 14 pixels \times 14 pixels, from an original size of 28 \times 28. Other limitations arise from the structure learning method used, which requires that all nodes are discrete and observed. That allows to find the optimal set of parents for each node separately. Therefore, digit images are binarised before learning the structure, with a threshold of half the maximum value of a pixel. The maximum number of parents per node is fixed to 7 in order to only learn the most useful links, and all parents of a node in time slice $t + 1$ belong to time slice t . For each node, the selected combination of parents is the one that corresponds to the highest BIC score according to the data.

An example of DBN structure from this model is presented in Fig. 1(a). For a given digit image, each pixel value is observed once, so the unrolled DBN has $t_{max} = 14$ time slices (one per column of an image), each time slice containing 14 evidence nodes (one per pixel of a column).

Model 2: Learned inter-slice links and hidden node. Fig. 1(b) shows the second kind of DBN structure tested. Evidence nodes are the same as previously, with the same inter-slice links and the same observations of columns of pixels. Each time slice contains an additional hidden discrete node linked to all evidence nodes of the slice, and to the hidden node in the next slice. The number of states of hidden discrete nodes is fixed to 13, as our experiments have shown no significative improvement for higher values.

Model 3: Coupled structure. The coupled structure (Fig. 1(c)) is obtained by linking two DBNs from the previous model, one observing a column of pixels per time slice, the second observing a line of pixels per time slice. The two hidden nodes are discrete and have 13 states. The inter-slice links between evidence nodes observing lines ($E_i^t, i = 15..28$) have been learned in the same way as those between evidence nodes observing columns ($E_i^t, i = 1..14$), but separately.

Model 4: Learned inter- and intra-slice links, and hidden node. This model (Fig. 1(d)) is based on Model 2, with additional intra-slice links between evidence nodes. Intra-slice links have been learned from the data in the same way as inter-slice links, i.e. using the BIC score, and a different set of intra-slice links has been learned for each class of data. To avoid cycles in the graph, intra-slice links were made to follow the constraint according to which E_i^t can only be a parent of E_j^t if $i < j$.

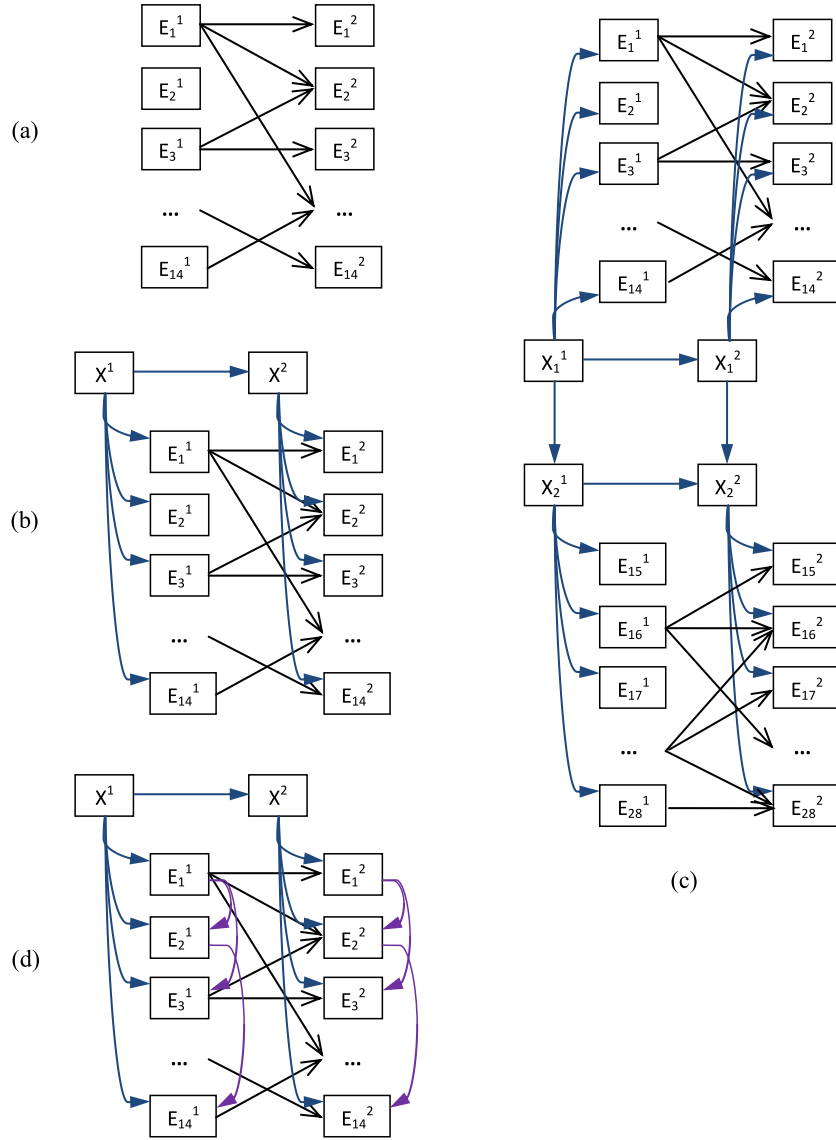


Fig. 1. The first two time slices of four DBN architectures. The complete unrolled DBNs have $t_{max} = 14$ time slices. E_i^t is the evidence node number i in time slice t , and X_j^t is the hidden node number j in time slice t (X^t if there is only one hidden node per time slice). (a) Model 1: A DBN made of evidence nodes only, with inter-slice links learned from the data (columns of pixels). (b) Model 2: Same evidence nodes as in (a), with one hidden node per slice, connected to every evidence nodes of the slice. (c) Model 3: A DBN observing columns and lines of pixels coupled through hidden nodes. (d) Model 4: Same as Model 2, completed with intra-slice links learned from the data. The links between evidence nodes are shown as an example, and they are different for each class of data.

3 Experiments and Results

The data we used is extracted from the MNIST database of handwritten digits [19], which contains a training set of 60000 examples of characters, and a test set of 10000 examples. Fig. 2 shows a few examples of characters from the MNIST database.



Fig. 2. Examples of characters extracted from classes 2, 4, 7 and 9 of the original MNIST database (size 28×28). Our experiments used images resized to 14×14 .

The experiments have been conducted with the BayesNet Toolbox for Matlab [20], which provides source code to perform numerous operations on BNs and DBNs. The DBN parameters are learned using the EM algorithm mentioned in section 2. Our training set was made by randomly extracting 500 samples of data per class (5000 samples in total) from the MNIST training set. For recognition, we used a total of 2000 samples of data randomly extracted from the MNIST test set. Images were resized to half their initial size to reduce the number of parameters in the models; image size reduction also results in information loss, which may hamper the learning and recognition tasks.

For each of the four models presented in section 2.2, experiments were conducted using discrete (binary) evidence nodes with binary images, and using Gaussian evidence nodes. For the latter case, pixel values have been scaled from the interval $[0,255]$ to $[0,1]$. During each of these eight experiments, one DBN was trained for each class of data, with the particular links previously learned from that class of data; then each digit of our test set was tested with the DBNs of the ten classes and assigned to the class whose DBN gave the highest log-likelihood.

The results are summarised in table 1. For each model, it can be seen that Gaussian evidence nodes provide an important improvement for this application compared to discrete ones. The hidden node in Model 2 resulted in significantly better results than those obtained with Model 1, but the additional intra-slice links (Model 4) brought only a marginal improvement compared to Model 2. The coupled architecture of Model 3 gave the best results for both kinds of evidence nodes, and reached an average recognition rate of 93.3% with Gaussian evidence nodes; the corresponding standard deviation of the recognition rate of the different classes is 3.4 percentage points. Digits from classes 1 and 0 were the most correctly classified by Model 3 (Gaussian evidence nodes) with recognition rates of respectively 98.3% and 97.2%, while classes 7 and 4 have the lowest recognition rates, respectively 87.5% and 89.1%, both of them being most of the time mistakenly labelled as 9.

In [11], tests of several HMM and DBN models on the original MNIST database are reported, with recognition rates ranging from 87.4% (Horizontal HMM), to

Table 1. Recognition rates (%) of handwritten digits

	Model 1	Model 2	Model 3	Model 4
Discrete evidence nodes	67.7	69.6	74.8	71.2
Gaussian evidence nodes	81.0	90.2	93.3	90.6

94.9% for the best DBN. The same study reported SVMs provided a recognition rate of 96.1%. However, our results are not readily comparable as they are obtained using reduced images (4 times less observations), which inevitably lowers the information available for learning and recognition.

4 Conclusion

In this paper we have presented a task of handwritten digit recognition using Dynamic Bayesian Networks. The observations of our models were columns and lines of pixels, with up to 28 evidence nodes and 2 hidden nodes per time slice. Four DBN models with different structures have been tested. The structure was partly learned from the training set of data, separately for each class of digit. We used a trade-off between complete automatic structure learning and manual heuristic search for an efficient structure, by learning inter- and intra-slice links between evidence nodes from data before incorporating them with hidden discrete nodes. For each model, experiments were conducted with binary and Gaussian evidence nodes, the latter giving much better results.

DBNs offer a great range of possibilities and there is much scope for exploration and improvement. As we have seen, this application leads to a certain disparity between the recognition rates of the different classes. Future work could be to perform a two-step learning, where the first step would be dedicated to detecting which class is the most difficult to classify, and the second step would focus on improving the DBN structure of the class identified during the first step, using not only the training data of that class but also the data of other classes in order to find the most discriminative links.

Acknowledgment

The authors wish to acknowledge the financial support for the research work supported by the MICIE project under the European Framework-7 Programme (Contract No: 225353).

References

1. Kelly, D.L., Smith, C.L.: Bayesian inference in probabilistic risk assessment—The current state of the art. *Reliability Engineering & System Safety* 94 (2008)
2. de Campos, L.M., Castellano, J.G.: Bayesian network learning algorithms using structural restrictions. *International Journal of Approximate Reasoning* 45 (2006)

3. Russell, S., Norvig, P.: *Artificial Intelligence, A Modern Approach*, 2nd edn. Prentice Hall, Englewood Cliffs (2003)
4. Murphy, K.P.: *Dynamic Bayesian Networks: Representation, Inference and Learning*, PhD dissertation, UC Berkeley, Computer Science Division (July 2002)
5. Mihajlovic, V., Petkovic, M.: *Dynamic Bayesian Networks: A State of the Art*, CTIT technical reports series, TR-CTIT-34 (2001)
6. Kao, H.-Y., Huang, C.-H., Li, H.-L.: Supply chain diagnostics with dynamic Bayesian networks. *Computers & Industrial Engineering* 49 (2005)
7. Daoudi, K., Fohr, D., Antoine, C.: Dynamic Bayesian networks for multi-band automatic speech recognition. *Computer Speech & Language* 17 (2003)
8. Huang, C.-L., Shih, H.-C., Chao, C.-Y.: Semantic analysis of soccer video using dynamic Bayesian network. *IEEE Transactions on Multimedia* 8 (2006)
9. Dielmann, A., Renals, S.: Automatic Meeting Segmentation Using Dynamic Bayesian Networks. *IEEE Transactions on Multimedia* 9 (2007)
10. Zajdel, W., Cemgil, A.T., Kröse, B.J.A.: Dynamic Bayesian Networks for Visual Surveillance with Distributed Cameras. In: Havinga, P., Lijding, M., Meratnia, N., Wegdam, M. (eds.) *EUROSSC 2006*. LNCS, vol. 4272, pp. 240–243. Springer, Heidelberg (2006)
11. Likforman-Sulem, L., Sigelle, M.: Recognition of degraded characters using dynamic Bayesian networks. *Pattern Recognition* 41 (2008)
12. Tsamardinos, I., Brown, L.E., Aliferis, C.F.: The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning* 65 (2006)
13. Pinto, P.C., Nagele, A., Dejori, M., Runkler, T.A., Sousa, J.M.C.: Using a Local Discovery Ant Algorithm for Bayesian Network Structure Learning. *IEEE Transactions on Evolutionary Computation* 13 (2009)
14. Rajapaksea, J.C., Zhoua, J.: Learning effective brain connectivity with dynamic Bayesian networks. *NeuroImage* 37 (2007)
15. Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, 2nd edn. Morgan Kaufman, Los Altos (1988)
16. Ghahramani, Z.: Learning Dynamic Bayesian Networks. In: Giles, C.L., Gori, M. (eds.) *IIASS-EMFCSC-School 1997*. LNCS (LNAI), vol. 1387, p. 168. Springer, Heidelberg (1998)
17. Friedman, N., Murphy, K., Russell, S.: Learning the Structure of Dynamic Probabilistic Networks. In: *Conference on Uncertainty in Artificial Intelligence, UAI 1998* (1998)
18. Friedman, N.: Learning Belief Networks in the Presence of Missing Values and Hidden Variables. In: *International Conference on Machine Learning* (1997)
19. LeCun, Y., Cortes, C.: The MNIST database of handwritten digits (1998), <http://yann.lecun.com/exdb/mnist/>
20. Murphy, K.P.: BayesNet Toolbox for Matlab, <http://people.cs.ubc.ca/~murphyk/Software/BNT/bnt.html> (last updated 2007)