CONTINUOUS SPEECH RECOGNITION USING STRUCTURAL LEARNING OF DYNAMIC BAYESIAN NETWORKS

Murat Deviren, Khalid Daoudi
INRIA-LORIA, Speech Group B.P. 101 - 54602 Villers les Nancy, France
e-mail: deviren,daoudi@loria.fr

ABSTRACT

We present a new continuous automatic speech recognition system where no a priori assumptions on the dependencies between the observed and the hidden speech processes are made. Rather, dependencies are learned form data using the Bayesian networks formalism. This approach guaranties to improve modelling fidelity as compared to HMMs. Furthermore, our approach is technically very attractive because all the computational effort is made in the training phase.

1 INTRODUCTION

First order Hidden Markov Models (HMM) are the most commonly used stochastic models in speech recognition. They are defined with a set of imposed conditional independence assumptions. Indeed, the observations are assumed to be governed by a hidden (unobserved) dynamic process. The associated independence assumptions state that the hidden process is first-order Markov, and each observation depends only on the current hidden variable. There is, however, a fundamental question regarding these dependency assumptions: are they realistic hypothesis for any kind of speech recognition task?

In [1], we proposed a methodology in which we do not make any *a priori* dependency assumptions. Rather, we give data a complete (but controlled) freedom to dictate the appropriate dependencies. In other words, we learn the dependencies between (hidden and observable) variables *from data*. The principle of this methodology is to search over all the possible "realistic" dependencies, and to chose the ones which best explain the data. This approach has the advantage to *guaranty* that the resulting model represents speech with higher fidelity than HMMs. Moreover, a *control* is given to the user to make a trade-off between modeling accuracy and model complexity. In addition, the approach is technically very attractive because all the computational effort is made in the *training* phase.

Our approach is based on the framework of *dynamic Bayesian networks* (DBNs). DBN theory is a generalization of Bayesian networks (BNs) to dynamic processes. Briefly, the Bayesian networks formalism consists in associating a directed acyclic graph to the joint probability distribution (JPD) $P(X)$ of a set of random variables $X = \{X_1, ..., X_n\}$. The nodes of this graph represent the random variables, while the arrows encode the conditional independencies (CI) which (are supposed to) exist in the JPD. The set of all CI relations, which are implied by the separation properties of the graph, are termed the *Markov properties*. A BN is completely defined by a graph structure S and the numerical parameterization Θ of the conditional probabilities of the variables given their parents. Indeed, the JPD can be expressed in a factored way as $P(X) = \prod_{i=1}^{n} P(X_i|\Pi_i)$, where Π_i denotes the parents of X_i in S.

The use of DBNs in speech recognition has gained a lot of interest in the last few years [2, 3, 4]. In this paper, we use the flexibility of this framework and instead of fixing *a priori* the structure of the acoustic models (as is done with HMMs), we build an "intelligent" system which works as follows. We feed the system using the observed data. Then, the system determines the structure S (i.e., the dependencies) and the parameters Θ which best represent the data. This strategy is known as *structural learning* in the BNs literature. In [1], we utilized this learning methodology for an isolated speech recognition task. In this paper, we present the application of this approach to continuous speech recognition. While inference is relatively easy in the isolated setting, it requires more attention in the continuous setting. We provide in particular a decoding algorithm which handles different model structures for different words in the vocabulary.

In the next section, we shortly introduce the dynamic Bayesian networks terminology. In section 3, we define the class of DBNs we use in our setting. Next, we briefly summarize the structural learning algorithm. In section 6, we describe the decoding algorithm for continuous speech recognition using DBNs. Finally, we illustrate the performance of our approach on a connected digits recognition task.

2 DYNAMIC BAYESIAN NETWORKS

A DBN encodes the joint probability distribution of a time evolving set $X[t] = \{X_1[t], \ldots, X_n[t]\}$ of variables. If we consider T time slices of variables, the DBN can be considered as a (static) BN with $T \times n$ variables. Using the factorization property of BNs, the joint probability density of $\mathbf{X}_1^T = \{X[1], \ldots, X[T]\}$ is written as :

$$P(X[1], \ldots, X[T]) = \prod_{t=1}^{T} \prod_{i=1}^{n} P(X_i[t]|\Pi_{it}) \qquad (1)$$

where Π_{it} denotes the parents of $X_i[t]$. In the BNs literature, DBNs are defined using the assumption that $X[t]$ is a Markov process [5]. In this paper, we relax this assumption to allow non-Markov processes and consider that the process $X[t]$ satisfies:

$$P(X_i[t]|\mathbf{X}_1^{t+\tau_f}) = P(X_i[t]|X[t-\tau_p], \ldots, X[t+\tau_f]) \qquad (2)$$

for some positive integers τ_p and τ_f. Graphically, the above assumption states that a variable at time t can have parents

in the interval $[t-\tau_p, t+\tau_f]$. However, care must be taken when dealing with boundary variables (see [1] for details).

From this perspective, it is easy to represent an HMM as a DBN. Indeed, the Markov properties (the dependency assumptions) of an HMM, are encoded by the graphical structure shown in Figure 1. Each node in this structure represents a random variable $X_h[t]$ or $X_o[t]$, whose value specifies the state or the observation at time t.

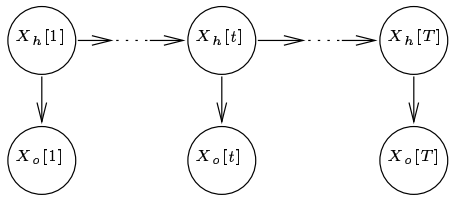

Figure 1: *HMM represented as a DBN*

3 STRUCTURE SEARCH CLASS

Learning the structure of DBNs, requires a search over a class of structures. Searching over all the possible DBN structures would be computationally infeasible. Therefore, we restrict ourselves to a small but rich set of structures that represents only "realistic" dependencies, in a physical and computational sense. The reader is referred to [1] for the reasoning behind the authorized dependencies.

Let $X[t] = \{X_h[t], X_o[t]\}$ be the set of hidden and observed variables at time t. The allowed dependencies are:

- The hidden variable at time t is independent of $\mathbf{X}_1^{t-\kappa-1}$ given the last κ hidden variables, for $t > \kappa$,

$$P(X_h[t]|\mathbf{X}_1^{t-1}) = P(X_h[t]|X_h[t-\kappa], \ldots, X_h[t-1]). \quad (3)$$

- The observation variable at time t is independent of all other variables given the hidden variables in the time window $[t-\tau_p, t+\tau_f]$, for some positive integers τ_p and τ_f,

$$P(X_o[t]|\mathbf{X}_1^T \setminus \{X_o[t]\}) = P(X_o[t]|X_h[t-\tau_p], \ldots, X_h[t+\tau_f]). \quad (4)$$

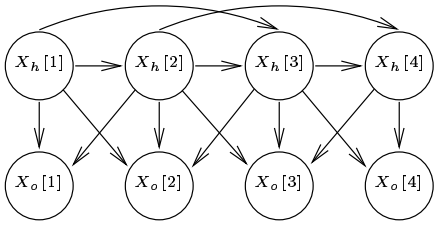

Figure 2: *DBN structure with* $(\kappa, \tau_p, \tau_f) = (2, 1, 1)$, $T = 4$

Hence, the search class of allowed DBN structures is defined by the triples (κ, τ_p, τ_f) for, $1 \leq \kappa \leq \kappa_{max}$, $0 \leq \tau_p \leq \tau_{p_{max}}$, $0 \leq \tau_f \leq \tau_{f_{max}}$, where $(\kappa_{max}, \tau_{p_{max}}, \tau_{f_{max}})$ is an upper bound which restricts the size of the search class. At the lower bound $(\kappa, \tau_p, \tau_f) = (1, 0, 0)$, the structure reduces to the standard first-order HMM (Figure 1) where, Eq.(3) defines the state transition probabilities and Eq.(4) defines the observation probabilities. Each triple (κ, τ_p, τ_f) specifies a DBN structure. As an example, for $(\kappa, \tau_p, \tau_f) = (2, 1, 1)$, we have the structure given in Figure 2.

If each discrete hidden variable $X_h[t]$ takes its values in the set of ordered labels [1] $I_v = \{1_v \ldots N_v\}$, and each observable variable has a conditional Gaussian density, the numerical parameterization of our DBNs is the following:

$$\begin{aligned} P_v(X_h[t] = j_v|\Pi_{ht}^v = \mathbf{i}_v) &= a_{\mathbf{i}_v j_v}[t], \ for\ j_v \in I_v \\ P_v(X_o[t]|\Pi_{ot}^v = \mathbf{i}_v) &\sim \mathcal{N}(\mu_{\mathbf{i}_v}[t], \Sigma_{\mathbf{i}_v}[t]). \end{aligned} \quad (5)$$

The (possibly) vector-index $\mathbf{i}_v$ is over all possible values of the variable's parents. For the specific class of structures we have considered, the parents are always hidden variables, thus, $\mathbf{i}_v$ is a point in the cartesian space I_v^m, where m is the number of parents of the variable under consideration.

In order to provide fair comparisons in the experiments, we compute the number of parameters required to define a DBN. Let M be the number of parameters required to encode the observation probability density. Then, for a left-to-right topology, the required number of parameters to encode the network is given by :

$$C = 2^\kappa N_v - \sum_{l=1}^{\kappa} l 2^{\kappa-l} + M \times [2^\beta N_v - \sum_{l=1}^{\beta} l 2^{\beta-l}]. \quad (6)$$

β is defined as $\beta = \tau_p + \tau_f$ (we assume that $\kappa, \beta \leq N_v$).

4 STRUCTURAL LEARNING

In the previous section, we defined a set of plausible DBN structures for speech modelling. In this section, we summarize the learning algorithm. The details of the algorithm are given in [1] and the references within.

Given a set of observations, our aim is to find the "optimal" model structure defined by the triple (κ, τ_p, τ_f), and the associated conditional probabilities which best explain the data. The optimality is considered in the sense that the likelihood of observations is maximum and the structural complexity is minimum. There are basically two scoring metrics which allow the evaluation of the degree to which a structure fits the data : the Bayesian Dirichlet (BD) metric, and Minimum Description Length (MDL) (or equivalently the Bayesian Information Criteria (BIC)) metric [6]. We use MDL metric, which penalizes complex structures based on the number of parameters used to encode the model. The MDL metric is defined as follows :

$$Score_{MDL} = \log P(D|\Theta, S) - \frac{\log L}{2} \sum_{i=1}^{n} ||X_i, \Pi_i|| \quad (7)$$

where D is the observation set, L is the number of examples (realizations) in D and $||X, Y||$ is defined as the number of parameters required to encode the conditional probability, $P(X|Y)$. The likelihood term can be computed using the JLO algorithm which is an efficient inference algorithm for Bayesian networks [7].

In order to find the optimal model, we use the structural EM (SEM) algorithm [8]. The algorithm starts with some initial structure and parameters. At each step, the expected scores of the candidate structures are computed according to the scoring metric (i.e. MDL). The structure that gives the maximum expected score is chosen as the next structure, and the parameters of this structure are updated with a standard parametric EM step. Updating the structure iteratively at

[1] The subscript v may be omitted in these notations. We use it however because in Section 5 we need to refer to word v in the vocabulary.

each step, we are guarantied to increase the score and to converge to a local maximum [8].

We initialize the algorithm with $(\kappa_{max}, \tau_{p_{max}}, \tau_{f_{max}})$ as the upper bound on the structure search space and use the HMM structure for the first iteration. This initialization guaranties that the resulting model will have a higher (or equal) fidelity, as compared to the HMM. The trade off between the complexity of the learning algorithm and the fidelity of the resulting model is controlled by the upper bound on the search space.

5 DECODING ALGORITHM

Let us assume that we are given a vocabulary V of $|V|$ words, and a DBN model for each word $v \in V$, with structure $(\kappa^v, \tau_p^v, \tau_f^v)$. The decoding problem is to identify the most likely sequence of words, given a speaker utterance. If all the DBN structures were the same, i.e. $(\kappa^v, \tau_p^v, \tau_f^v) = (\kappa, \tau_p, \tau_f),\ \forall\ v$, then an efficient solution to our decoding problem is the following. The basic idea is to build a state-augmented "super" DBN model (with the structure (κ, τ_p, τ_f)) which represents all the words in the vocabulary. In this super model, each variable $X_h[t]$ takes now its values in the set $I = \bigcup_{v \in V} I_v$.

To complete the definition of this super model, we need to specify the conditional probabilities $P(X_i[t]|\Pi_{it} = \mathbf{i})$, for $i \in \{h, o\}$. These are obtained from individual word models and the language model. First recall that Π_{it} is always a subset of hidden variables. When the configuration $\mathbf{i}$ of Π_{it} corresponds to the same word v, i.e. $\mathbf{i}$ takes its values in I_v^m, then the conditional probabilities are given by the inside-word parameters:

$$P(X_h[t] = j_v|\Pi_{ht} = \mathbf{i}_v) = a_{\mathbf{i}_v j_v}[t],\ \forall\ \mathbf{i}_v \in I^{\kappa^v}, j_v \in I. \quad (8)$$

$$P(X_o[t]|\Pi_{ot} = \mathbf{i}_v) \sim \mathcal{N}(\mu_{\mathbf{i}_v}[t], \Sigma_{\mathbf{i}_v}[t]),\ \forall\ \mathbf{i}_v \in I^{\tau_p^v + \tau_f^v + 1}. \quad (9)$$

However, there are also some configurations where some parents take values corresponding to different words. These specify the parameters for word transitions. Note that, we only allow transitions from the last state of a word to the first state of another, and assign zero probability for all other configurations. We use the language model for these transition probabilities of the hidden variable. For the observable variable, the emission probabilities in the word transitions are specified using the word model that is indicated by the configuration of $X_h[t]$. When $X_h[t] = N_v$ and $X_h[t+1] = 1_{v'}$, we assume that the emission is from word v such that the state sequence ends at the last state. Similarly, when $X_h[t-1] = N_{v'}$ and $X_h[t] = 1_v$, we assume that the emission is from word v such that the state sequence starts at the first state of word v. Hence, the conditional probabilities for the transitions between two (not necessarily different) words v and v' are given as:

$$P(X_h[t] = 1_v|\Pi_{ht} = \mathbf{i}_{v'}) = P(v|v')$$
$$\forall\ \mathbf{i}_{v'} \in I^{\kappa^{v'}} \text{ s.t. } \Pi_{ht} \ni X_h[t-1] = N_{v'}. \quad (10)$$

$$P(X_o[t]|\mathbf{i}) = \begin{cases} P_v(X_o[t]|(1_v \ldots 1_v, i_{t+1} \ldots i_{t+\tau_f})) \\ \quad \text{for } i_t = 1_v \text{ and } (i_{t+1} \ldots i_{t+\tau_f}) \in I_v^{\tau_f}. \\ P_v(X_o[t]|(i_{t-\tau_p} \ldots i_{t-1}, N_v \ldots N_v)) \\ \quad \text{for } i_t = N_v \text{ and } (i_{t-\tau_p} \ldots i_{t-1}) \in I_v^{\tau_p}. \\ 0 \text{ otherwise} \end{cases} \quad (11)$$

where $P(v|v')$ defines the language model. Notice that, for a left to right topology, all probabilities are set to zero for $i_t \neq i_{t-1}, i_{t-1} - 1$.

Once such a super model is constructed, the decoding can be performed using the Dawid algorithm [9] which allows the identification (with the same time complexity as the JLO algorithm [7]) of the most likely sequence of hidden states, given observations. It is important to note that if the DBN structures reduce to HMM, then this technique is equivalent to Viterbi decoding [10].

In general, after performing a structural learning, the resulting DBN structures would not be the same for all the words. In this case, we cannot directly use the technique described above, because we do not have, at hand, a unique structure to represent all the words. Indeed, in the augmented model when $X_h[t]$ takes a value j_v corresponding to a word v, the independence assertions should be identical to those imposed by the structure $(\kappa^v, \tau_p^v, \tau_f^v)$. In other words, we have to encode *asymmetric* independence assertions in the sense that variables are independent for some but not necessarily for all of their values. We present, in the following, a technique to do so which relies on encoding the asymmetric assertions in the numerical parameterization. The basic idea is to build a "maximal" DBN model and use a special parameterization to encode the asymmetry.

We find the "maximal" DBN structure $(\kappa^m, \tau_p^m, \tau_f^m)$ that is capable to encode all the dependencies represented in the learned word models as follows :

$$\kappa^m = \arg\max_{\kappa, v}(\kappa, \tau_p, \tau_f)_v,\ \tau_p^m = \arg\max_{\tau_p, v}(\kappa, \tau_p, \tau_f)_v$$
$$\tau_f^m = \arg\max_{\tau_f, v}(\kappa, \tau_p, \tau_f)_v \quad (12)$$

Then, for each word, we construct an equivalent DBN with this maximal structure that encodes the same JPD as the original word DBN. Although the Conditional Independence (CI) assertions of the maximal structure is different from the word DBN, we exploit the Markov properties of the word DBN in specifying the conditional probabilities of the equivalent maximal DBN. This comes back to setting several of them to be equal to those given by the original model. Precisely,

$$P_{mv}(X_i[t]|\Pi_{it}^{mv} = (\mathbf{i}_v, \mathbf{j})) = P_v(X_i[t]|\Pi_{it}^v = \mathbf{i}_v)$$
$$\text{for all } \mathbf{i}_v \in \Pi_{it}^v, \mathbf{j} \in \Pi_{it}^{mv} \setminus \Pi_{it}^v \quad (13)$$

Now that we have a set of DBNs with the same structure, we can construct a super DBN as described above. Note that, since we do not violate the different CI assertions of individual word models in this procedure, the CI assertions encoded in the resulting super DBN are asymmetric.

Obviously, the extra parameterization in the maximal model causes unnecessary computation while performing inference. A more efficient way to encode asymmetric independence assertions is to use Bayesian multinets (or similarity networks). The theory of Bayesian multinets generalizes Bayesian networks by using a representation employing multiple networks and thereby allowing asymmetric independencies [11]. Such a representation may substantially reduce the inference complexity, still, it yields exactly the same inference results as the ones obtained using the representation described above. Since our goal, in this paper, is to show the potential of structural learning and given the lack of space, we do not get into the details of the Bayesian multinets approach and leave it to a future work.

6 EXPERIMENTS

In this section, we compare[2] the performances of DBN models to standard HMMs. Our experiments are carried out on the Tidigits database. In learning, we only use the isolated part of the training database where each speaker utters 11 digits ('o',0, 1, ... 9) twice. As an initial step, we segment the silence regions in the training corpus. We perform isolated training for all HMM and DBN models based on this initial segmentation. In all experiments, the silence is modelled with a single state, single Gaussian HMM. In tests, we use the full (test) database in which 8636 sentences are uttered, each sentence contains between 1 and 7 digits. The size of each observation vector is 35, consisting of 11 MFCCs (energy dropped), 12 Δ, and 12 $\Delta\Delta$. The covariance matrix is assumed diagonal. We use a left-to-right transition topology and a uniform language model, i.e., $P(v|v') = \frac{1}{12}$ ($|V| = 12$). In the structural learning algorithm, for all digits, the upper-bound on the search class is $(\kappa_{max}, \tau_{p_{max}}, \tau_{f_{max}}) = (2, 1, 1)$.

(κ, τ_p, τ_f)	N=4	N=5	N=6
(1,0,0)			'o',0,2,7
(1,0,1)	'o',2,4,7,8,9	'o',1,4,5,9	3,6,8
(1,1,0)	0,1,3,5,6	0,2,3,6,7,8	1,4,5,9

Table 1: *Results of SEM algorithm.*

We show experiments using different numbers of states, i.e. different values of N where we set $N_v = N \; \forall \; v$. In Table 1, we show the results of the structural learning algorithm. When $N \geq 7$, our system yields a standard HMM as the best model for all digits. This shows that when an HMM is enough to model data, our system recognize it. The recognition scores are given in Table 2, for $N = 4, 5, 6$, the maximal model structure is $(\kappa^m, \tau_p^m, \tau_f^m) = (1, 1, 1)$. Our system largely outperforms the HMM system, particularly when $N = 4$. This remarkable increase in the performance can be explained as follows. Since the number of hidden states is small, the HMM system introduces a lot of insertions. Using our system, the learned models all involve context dependencies. It is well known that modelling context, improves duration modelling. Therefore, even though the number of states is still small, our system provides a better digit duration modelling, and thus the total number of insertions is drastically reduced. However, to make fair comparisons, we should compare our system to an HMM, with an equivalent number of parameters. To do so, we compare our system to an HMM with the same number of states, but using a mixture of 2 Gaussians as the observation probability density. The average number of parameters C_{av} of each system is computed by averaging the quantity C defined by Eq. (7) over all digit models [3]. For $N = 4, 5$, our system ($C_{av} = 497,\ 639$) performs still better than HMM_{2G} (which actually uses even more parameters than our system, $C_{av} = 567,\ 709$). For $N = 6$, even though our system uses much less parameters ($C_{av} = 653.6$) than HMM_{2G} ($C_{av} = 1551$), both systems yields the same scores approximately.

As a final remark on these experiments, note that our system allows learning of non-Markovian processes (the DBN (1,0,1) is non-Markov, for instance). This is a major advantage with respect to HMMs. Indeed, by doing so, our system is able to take into account the well known *anticipation* phenomenon which occurs in the speech production mechanism. These experiments show the power of the DBN approach to model both acoustical and phonemical aspects of speech with higher fidelity than HMMs.

[2]Very good scores can be obtained on this database using Gaussian-mixture HMMs and adjusted parameters. Our goal here is not to tune the parameters in order to achieve the highest performances. Rather, we want to provide fair comparisons using baseline systems. We believe that this way we have a fair initial judgment on the capacities of each system.

[3]For 1 Gaussian $M = 70$, for 2 Gaussians $M = 140$.

N	HMM(N)	DBN(N)	$\text{HMM}_{2G}(N)$
4	20.57	63.22	46.32
5	60.92	79.86	77.99
6	77.93	84.47	84.44

Table 2: *Digit recognition accuracies (%).*

7 CONCLUSION

We used the DBN framework to construct acoustic models that are capable to learn the dependency structure of the hidden and observed speech processes. We presented a practical methodology to use such models in continuous speech recognition. The approach allows the use of different model structures for different units in the vocabulary. We showed that using our models, we are able to achieve better recognition scores as compared to equivalent HMMs.

References

[1] M. Deviren and K. Daoudi, "Structural learning of dynamic bayesian networks in speech recognition," in *Eurospeech*, 2001.

[2] J. A. Bilmes, "Dynamic bayesian multinets," in *UAI*, 2000.

[3] K. Daoudi, D.Fohr, and C. Antoine, "Continuous multi-band speech recognition using bayesian networks," in *ASRU*, 2001.

[4] G. Zweig, *Speech Recognition with Dynamic Bayesian Networks*, Ph.D. thesis, University of California, Berkeley, Spring 1998.

[5] N. Friedman, K. Murphy, and S. Russell, "Learning the structure of dynamic probabilistic networks," in *UAI*, 1998.

[6] David Heckerman, "A tutorial on learning with bayesian networks," Tech. Rep. MSR-TR-95-06, Microsoft Research, Advanced Technology Division, March 1995.

[7] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen, "Bayesian updating in recursive graphical models by local computations," *Computational Statistics and Data Analysis*, vol. 4, pp. 269–282, 1990.

[8] N. Friedman, "Learning belief networks in the presence of missing values and hidden variables," in *Int. Conf. Machine Learning*, 1997.

[9] A.P. Dawid, "Application of a general propagation algorithm for probabilistic expert systems.," *Statistics and Computing*, , no. 2, pp. 25–36, 1992.

[10] P. Smyth, D. Heckerman, and M. I. Jordan, "Probabilistic independence networks for hidden markov probability models," *Neural Computation*, vol. 9, no. 2, pp. 227–269, 1997.

[11] D. Geiger and D. Heckerman, "Knowledge representation and inference in similarity networks and bayesian multinets," *Artificial Intelligence*, vol. 82, pp. 45–74, 1996.