Computer Speech and Language (1998) **12,** 23–39

Efficient training of high-order hidden Markov models using first-order representations

J. A. du Preez

University of Stellenbosch, Private Bag X1, Matieland 7602, South Africa

Abstract

We detail an algorithm (ORED) that transforms any higher-order hidden Markov model (HMM) to an equivalent first-order HMM. This makes it possible to process higher-order HMMs with standard techniques applicable to first-order models. Based on this equivalence, a fast incremental algorithm (FIT) is developed for training higher-order HMMs from lower-order models, thereby avoiding the training of redundant parameters. We also show that the FIT algorithm results in much faster training and better generalization compared to conventional high-order HMM approaches. This makes training of high-order HMMs practical for many applications.

1. Introduction

Due to their high computational complexity, higher-order hidden Markov models (HMMs) have found little application in existing speech processing systems. For example, in an ergodic Rth-order HMM with N states, there are N^R transition probabilities. This large number of parameters also requires a very large database for adequate training, which compounds this problem. In addition to these difficulties, specialized extended algorithms are necessary for each specific order of HMM.

In previous work, both the Viterbi (He, 1988) and the Baum-Welch (Kriouile, Mari & Haton, 1990) algorithms have been extended to second-order models. The enhanced duration modelling of second-order HMMs, compared to first-order models, improves the accuracy obtained in digit and letter recognition experiments (Mari & Haton, 1994; Mari, Fohr & Junqua, 1996; Mari, Haton & Kriouile, 1997). Mari & Haton (1994) and Mari *et al.* (1997) note the equivalence between second-order and first-order HMMs; however, they do not discuss higher-order models, or provide algorithms for obtaining equivalent models. They conclude that the dramatic expansion in the number of states does not warrant the use of such a technique and instead use extended second-order Viterbi and Baum-Welch algorithms. Using a sparse representation they avoid

0885–2308/98/010023 + 17 $25.00/0/la970037

calculations involving redundant transition probabilities. We refer to these algorithms as the extended approach.

In Section 3 we detail an algorithm that *transforms higher-order HMMs to fully equivalent first-order versions* (Appendix A provides a proof of this). This is highly advantageous as *standard first-order HMM training and processing algorithms can be used* to process such models. We also use a sparse representation when processing the equivalent first-order models, thereby achieving the same computational efficiency as in the extended approach. To further enhance training efficiency, we also detail a new fast incremental training (FIT) algorithm in Section 4. Compared to prior work, FIT substantially reduces computational requirements. Section 5 applies this work to a limited automatic language recognition (ALR) task.

2. Definitions, notation and some background

$\mathbf{X}_1^L = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_\ell, \cdots \mathbf{x}_L\}$ is an observation sequence or string of length L, which must be matched to the HMM. An HMM M is defined as a set of N conventional emitting states, as well as an initial and terminal state that are so-called non-emitting or null states, yielding a total of $N+2$ states. The symbols S and Q are variables taking on the index value of the state under consideration. In normal HMMs this index value is a scalar, such as $Q=i$ (single indexed). In the transformed models developed in Section 3, composite values such as $Q=(i,j)$ (composite indexed) are used for intermediate indexing. This composite indexing is used to track state histories. Subscripts are used to specify the time at which a certain state occurs. An expression such as $S_\ell = j$ will indicate the occurrence of the jth state at time ℓ. A sequence of states occurring from time m up to time n is denoted by S_m^n. Time indexes lower than 1 or higher than L are not associated with physical time as is measured by the indexes of the observation string, but rather indicate null states preceding or following the input string.

States are coupled by transitions, each with probabilities describing its likelihood of occurring. The initial state will have no transitions entering it and the terminal state will have no transitions leaving it. Transition probabilities are indicated by the symbol a, with subscripts to index the states involved. To distinguish between the transition probabilities of the original higher-order models and its transformed equivalents, a prime ($'$) symbol is used for the former. For example, a'_{ijk} is the symbol used in the original second-order HMM to indicate the probability of moving from state j to state k, given that the preceding state was i.

Each emitting state has an associated probability density function (pdf.) $\mathrm{f}(\mathbf{x}|S,M)$ that quantifies the similarity between a feature vector $\mathbf{x}$ and the state. The ith pdf. is identified as f_i in the diagrams. The null states, indicated in state diagrams by dashed circles, do not have pdf.s associated with them. Transitions to them do not consume a time-step; this makes them useful tying points for groups of states, enabling the use of single initial and termination states whilst maintaining the functionality of the multiple case. This simplifies the representation by making the customary inclusion of extra parameters to indicate multiple initial and termination states unnecessary. In the following discussion, extensive use is made of states that share pdf.s (Bahl, Jelinek & Mercer, 1983; Lee, 1989). These are referred to as *tied pdf.s*.

The match between $\mathbf{X}_1^L$ and M is quantified by the likelihood $\mathrm{f}(\mathbf{X}_1^L|M)$. To make its calculation tractable, certain simplifying assumptions are necessary:

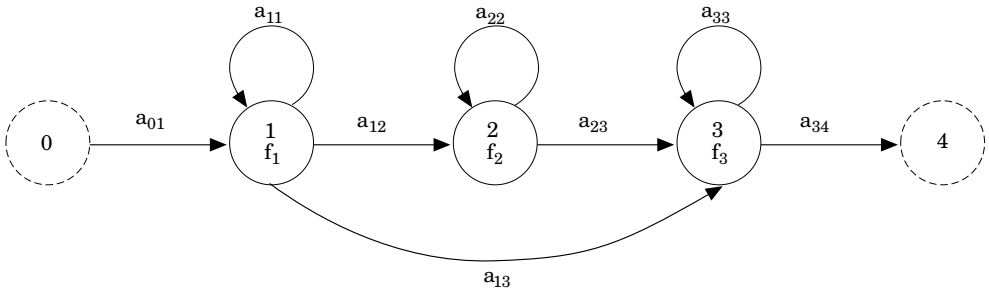

Figure 1. Base first-order HMM.

Assumption 1

The observation independence assumption states that

$$f(\mathbf{x}_\ell|\mathbf{X}_1^{\ell-1}, Q_0^\ell, M) = f(\mathbf{x}_\ell|Q_\ell, M). \tag{1}$$

This means that the likelihood of the ℓth feature vector is dependent only on the current state and is therefore otherwise unaffected by previous states and feature vectors. This assumption is not affected by the order of the HMM.

Assumption 2

By definition, an HMM includes the Markov order assumption

$$P(Q_\ell|Q_0^{\ell-1}, \mathbf{X}_1^{\ell-1}, M) = P(Q_\ell|Q_{\ell-R}^{\ell-1}, M), \tag{2}$$

where R is known as the order of the HMM.

This assumption states that any states or features, other than the immediately preceding R states, do not affect the probability of occurrence of the next state.

The vast majority of applications use $R=1$, resulting in a first-order HMM (HMM1) of which an example is shown in Figure 1. The probability of jumping to a specific state at the next time step is dependent only on the state that is occupied at the current time. Therefore, only a single transition probability occurs on the link between any two states. These probabilities are often presented as a matrix, and algorithms used to calculate the required likelihood only need to keep track of the behaviour of states one time step earlier (Poritz, 1988).

Since a transition to a next state is dependent only on the current state, more subtle restrictions on the allowable combinations of states are not effectively modelled (He, 1988). In addition, the self-loops on the emitting states are poor duration models in most practical applications (Levinson, 1986). The richer modelling capability resulting from increasing the order of the model can mitigate these difficulties (He, 1988; Mari

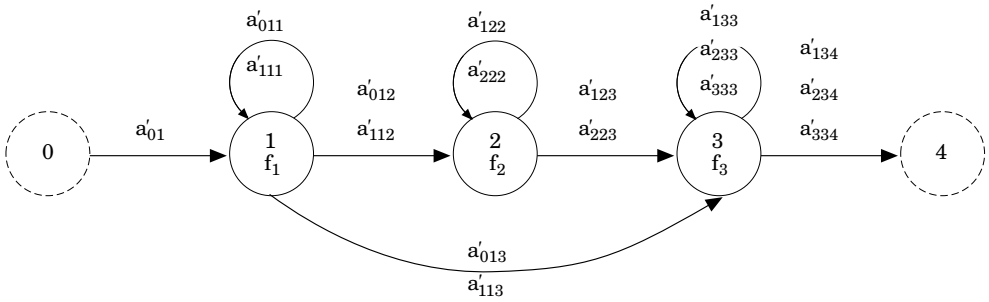

Figure 2. Second-order version of the HMM in Fig. 1.

et al., 1997). In Figure 2 the second-order version (HMM2) of the previous example is given. More precise modelling is now possible, but it comes at the cost of an increased number of transition probabilities. Note that, unlike the HMM1, both the previous and the current state determine the correct choice amongst the number of alternative probabilities on a given transition. Typically these probabilities are represented in a cubic structure, and the calculation of $f(\mathbf{X}_1^L|M)$ requires specialized algorithms to track the longer history of states (Kriouile *et al.*, 1990).

3. Transforming a *R*th-order to an equivalent first-order HMM

In this discussion, the Rth-order HMM is referred to as M_R, while the equivalent $R-1$th-order version is M_{R-1}. States in M_R are denoted by Q while the variable S is used for those in M_{R-1}. We assume that M_R has N emitting states, augmented by initial ($Q=0$) and terminal ($Q=N+1$) null states. States Q are denoted by single indexes whereas S use (intermediate) composite indexing.

3.1. Order reducing (ORED) algorithm

(1) Create the states of M_{R-1}:
   (a) Create the initial null state $S=0$.
   (b) For every state $Q=i$, $i=0\cdots N$ with a transition leading to state $Q=j$, $j=1\cdots N+1$, create a new state $S=(i, j)$.
   (c) If the terminal state $Q=N+1$ has multiple transitions entering it, create a new terminal null state $S=N+2$.

(2) Allocate pdf.s:
   All states $S=(j, k)$ shares, via tied pdf.s, the same pdf. as state $Q=k$. That is, they are all either null states, or $f(\mathbf{x}|S=(j, k),M_{R-1})=f(\mathbf{x}|Q=k, M_R)$.

(3) Allocate transitions and their probabilities:
   (a) Create a transition between state $S=0$ and each of the created states $S=(0, i)$. Associate the probability $a_{0(0,i)}=a'_{0i}$ with this transition.
   (b) Let $i_1\cdots i_m$ represent any sequence of indexes with length $m\leq R-2$. Create a transition between every existing state $S=(i, j)$ and $S=(j, k)$. The associated transition probabilities are $a_{(i_1,i_2)(i_2,i_3)\cdots(i_m,i)(i,j)(j,k)}=a'_{i_1\cdots i_m ijk}$.
   (c) If a new terminal state $S=N+2$ has been created, create a unit probability transition $a_{(i,N+1)(N+2)}=1$ between every state $S=(i,N+1)$ and $S=N+2$.

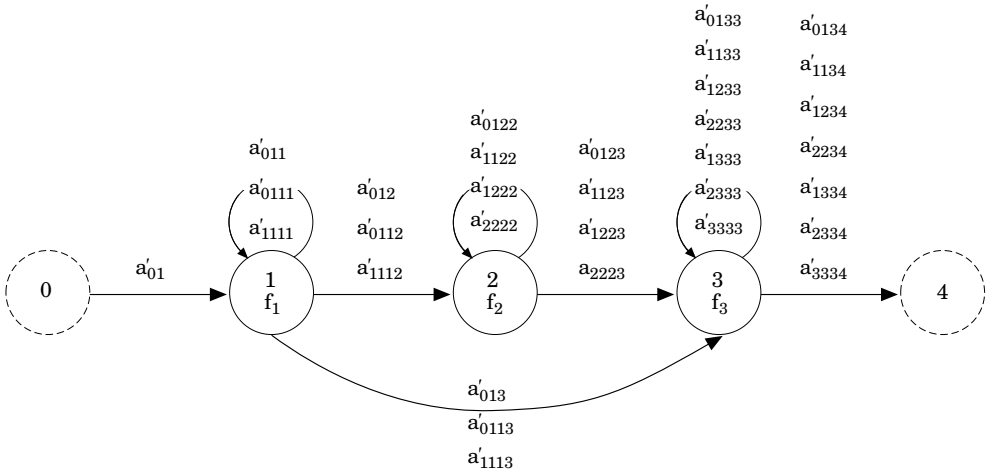

Figure 3. Third-order version of the HMM in Fig. 1.

(4) Iterate:
   (a) Rename all the composite state indexes to unique single indexes, and modify the indexes in the transition probabilities accordingly. The resulting model is the $R-1$th-order equivalent of the original Rth-order model.
   (b) All of the above steps can be repeated until all transition probabilities are subscripted by only two state indexes. By definition, this is a first-order model, shown in Appendix A to be equivalent to the original higher-order model.

3.2. Notes on the ORED algorithm

Steps 1(a) and (c) ensure that the model has unique initial and terminal states. Step 1(b) creates the necessary states to ensure that an extra step of state history is made explicit in the new model. Since the pdf. associated with each state is unaffected by the order of the model, step 2 uses tied pdf.s to ensure that the status quo is maintained when the new model is trained. The transitions leaving directly from the initial state necessarily is first-order and is handled separately in step 3(a). Step 3(b) inserts the higher-order transition probabilities. When the base model has order higher than two, there are multiple probabilities associated with each transition. The sequence $i_1 \cdots i_m$, $m \leq R-2$ is used to enumerate those probabilities. Step 3(c) ensures a single termination state, thereby keeping the representation cleaner. The superfluous null states leading to the termination state $S=N+2$ may (optionally) be eliminated. Recursion is used in step 4 for further decreasing of the order. Step 4(a) does the necessary renaming to the notation assumed by step 1. The use of single state indexes in the base model keeps the formulation simpler. Another attractive alternative would be to use composite indexes indicating the applicable history of states according to the order of the model. Step 4(b) is the actual recursion.

3.3. Example

In this section the ORED algorithm is applied to the third-order model of Figure 3. Figure 4 shows the resulting second-order equivalent to this model. The composite

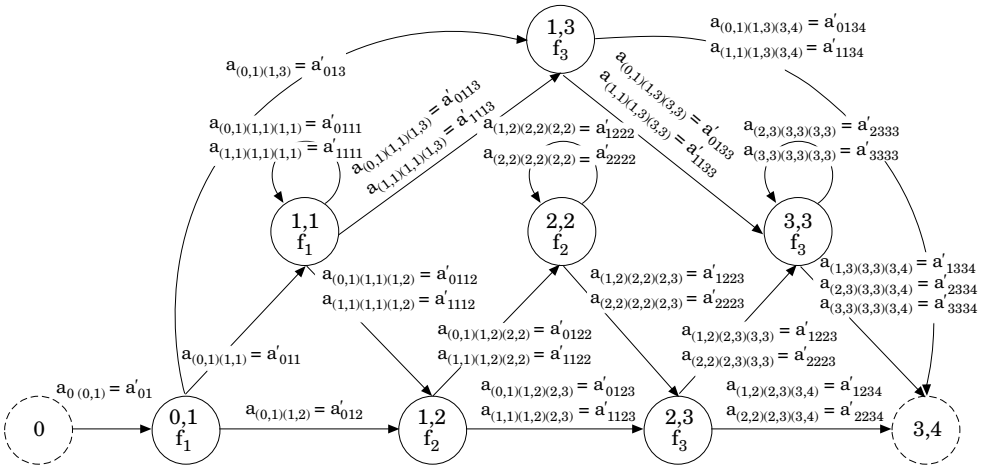

Figure 4. The second-order equivalent of the third-order HMM in Fig. 3, using the intermediate composite state indexing scheme. If the first index is dropped in cases where a has three and a' has four indexes, thereby leaving only single probabilities on each transition, this would also be the first-order equivalent of Fig. 2.

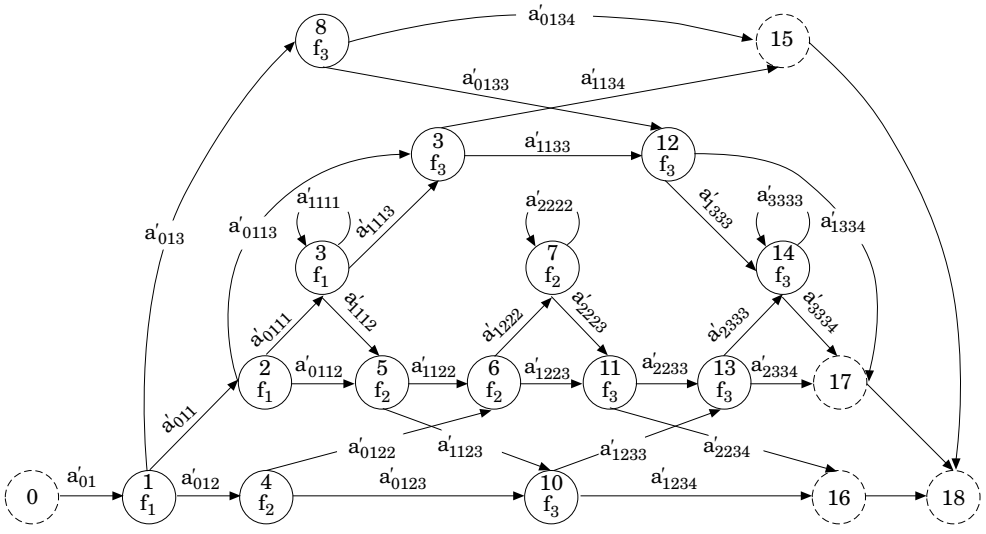

Figure 5. The first-order equivalent of the HMM in Fig. 3. The states of the original model in Fig. 3 correspond to states here with the same pdf. (f_i).

state indexes have been retained for illustrative purposes. In Figure 5 the first-order equivalent of this model is shown. As noted from these diagrams, the resulting models have a rich and aesthetically pleasing structure. (Other configurations also result in elegant structures and the reader is invited to sketch the first-order equivalents of small

second and third-order ergodic models.) Also note that, whereas the longest path without self-loops in Figure 1 occupies three physical time steps, it is six time steps for the second-order model and nine time steps for the third-order model. The shortest paths, of course, do not differ from the original first-order model. Explicit modelling of paths of different lengths results in more precise duration modelling.

4. Fast incremental training (FIT) of higher-order HMMs

As noted in Section 1, Kriouile *et al.* (1990) developed enhanced training algorithms for second-order HMMs. Using ORED, the HMM can also be trained by using standard first-order optimization algorithms. Due to the large number of parameters in higher-order models, they are plagued by two problems. These are: computationally very expensive training, and poor generalization to previously unseen data. To address this, we develop a fast training technique in the following section. We show in Sections 4.4 and 5 that the FIT algorithm, while directly addressing computational issues, also shows beneficial generalization behaviour. Interpolation and other techniques from language modelling (Jelinek, Mercer & Roukos, 1992), although not addressed here, are expected to provide additional robustness against insufficient amounts of training data.

4.1. Basis for Fast Incremental Training

Experience shows that in many practical situations involving non-trivial models, a large percentage of transitions disappear during training. For many problems, considerable training effort is therefore expended on estimating parameters that will eventually become zero. From Figures 1, 2 and 3, we see that a single transition probability in the $R-1$th-order model is simply being replaced by a set of probabilities in the Rth-order model. We now investigate the target transition probabilities of a first-order model (such as M_1 in Fig. 1) when it is trained with data that have been generated by the corresponding second-order model (such as M_2 in Fig. 2) (see Appendix B for a more formal analysis). Training algorithms, such as Baum-Welch re-estimation, estimate transition probabilities by counting the expected number of transitions on each of the links leaving from a state (Poritz, 1988). Consider a transition in M_2 from state $Q_{\ell-1}=j$ to $Q_\ell=k$ occurring with non-zero probability. If M_1 was used in the place of M_2, this same transition will be observed with a potentially different, but still non-zero probability. This suggests that a first-order model can be trained to determine which sets of second-order transition probabilities are viable. Redundant second-order probabilities can then be avoided during subsequent training.

4.2. The FIT algorithm for the training of fixed-order HMMs

(1) Set up a first-order HMM for the application at hand.
(2) Run the training algorithm on the first-order model. Non-viable transitions should disappear.
(3) Convert the optimized first-order model to a second-order model. Let A_j be the set of states that directly precedes state $S=j$. Replace the probability a_{jk} that joins the states $S=j$ and $S=k$ by the multiple probabilities a_{ijk} where $i\in A_j$. Initialize these probabilities with the first-order values, $a_{ijk}=a_{jk}$, $i\in A_j$. This conversion will

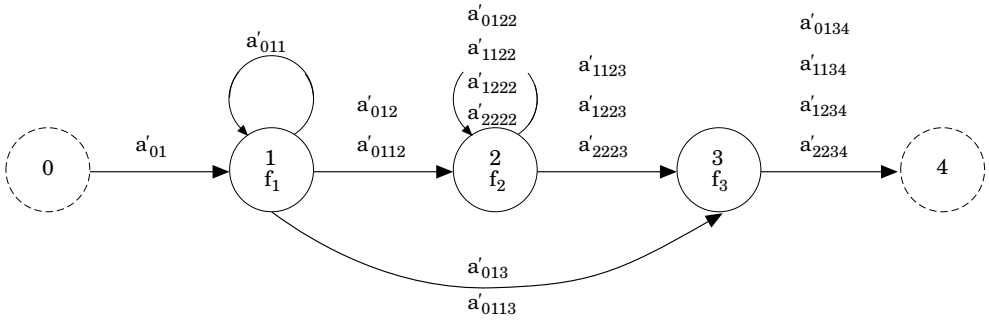

Figure 6. Model used for generating simulated data.

increase the number of transition parameters. If any transitions disappeared while training the first-order model, they will not propagate to the second-order model, thus avoiding the training of this larger number of second-order transition probabilities. To benefit from this a sparse transition matrix representation must be used.

(4) Use the ORED algorithm to create a first-order equivalent of this model. Initially this model will match an unknown observation string with the same likelihood as the original (trained) first-order model.

(5) Now, by repeating the algorithm from step 2, train this model. This will refine the transition probabilities to their required higher-order values. As was previously mentioned, an Rth-order model simply extends the memory of an $R-1$th-order model by one time step. Therefore this process can be repeated to train even higher order models, bearing in mind that deficiencies might arise from inadequate quantities of training data.

4.3. Example of training via the ORED and FIT algorithms

This section illustrates the above by recovering a high-order HMM from simulated data. This is done by using both the extended/ORED algorithm as well as the FIT algorithm. The model of Figure 6 was used to generate simulated data. This model was determined by removing some of the transition probabilities from the model of Figure 3. These removals were chosen to show specific effects at different levels (iterations) of the FIT algorithm. Firstly, all transitions involving a transition of state $S=3$ to itself were removed (a_{33} in Fig. 1). This is a *first-order effect* that eliminates four probabilities in the second-order model, and ultimately eliminates nine probabilities in the third-order model (all a' in Figs 2 and 3 with two or more 3's in the subscript). Next all transitions from state $S=1$ to itself, given that the previous state was also $S=1$, have also been removed (a'_{111} in Fig. 2). This is a *second-order effect* that ultimately eliminates four probabilities in the third-order model (all a' in Fig. 3 with three or more 1's in the subscript). Finally, a *third-order* transition probability, namely a'_{0123}, has also been removed. The remaining probabilities were fixed at arbitrarily chosen values. Two-dimensional data are assumed. The three diagonal Gaussian pdf.s of the emitting states were centred on the points (0, 0), (0, 1) and (1, 1). In two sub-experiments the standard deviations used in the pdf.s were varied. In the first, a small standard deviation

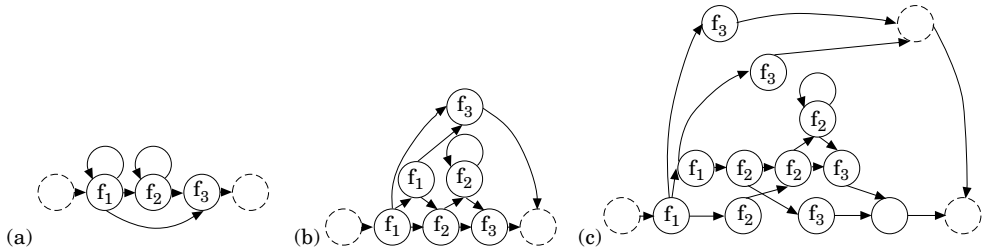

Figure 7. Model structures after (a) first, (b) second and (c) third-order stages of the FIT algorithm.

TABLE I. Mean deviation between trained and actual parameters

SD σ	Extended/ORED training	Incremental training
0·20	0·015	0·015
0·33	0·307	0·017

($\sigma = 0{\cdot}2$) produced little overlap between the pdf.s. The other experiment used a wider standard deviation, namely $\sigma = 0{\cdot}33$, giving considerable overlap. For each of these sub-experiments 1000 different strings, with randomly varying lengths as dictated by the model, of simulated feature vectors were drawn. Extended/ORED training was based on the structure of Figure 5 as initial configuration, whereas the incremental approach was initially based on Figure 1. The initial transition probabilities for the self-loops were fixed at 0·8 and the remainder was equally divided between the other transitions. Initial values for the pdf.s were determined from vector quantization (Gray, 1984).

Figure 7 shows the model topologies as they evolved from stage to stage in the incremental training process. Both sets of standard deviations yielded identical model structures. Compare this figure to those of Figures 3, 4 and 5. *The transitions that disappear during each stage of the incremental training correspond exactly to the model used to generate the data with.* Due to the presence of local optima this will not necessarily be so in all cases. Two different quantities were calculated with which to judge the finer differences between the models. The transitions and mean values in the models are roughly of comparable magnitude. The first quantity used measures the average difference between these trained parameters and their actual underlying values. Using the narrow pdf.s ($\sigma = 0{\cdot}2$) both the ORED and FIT approaches converged to the correct structure with identical parameters. As can be seen from Table I, the trained parameters closely matched the true underlying values. With the wider pdf.s ($\sigma = 0{\cdot}33$), the extended/ORED approach converged to a sub-optimal structure, clearly reflected in the rather large deviation from the true values. In spite of this, the total likelihood

TABLE II. Total likelihood $\log[f(\mathbf{X}_1^L|M)]$ after training

SD σ	Extended/ ORED	Incremental training order (FIT)				
		1	2	3	4	5
0·20	−503·74	−853·28	−649·17	−503·74	−501·46	−501·35
0·33	−5703·66	−5944·01	−5700·87	−5611·79	−5609·52	−5609·25

reported in Table II is reasonable and compares with the (correct) second-order approximation of the incremental approach. This is the typical behaviour of convergence towards a local optimum, showing that the solution from the FIT algorithm may or may not coincide with that of the extended/ORED approach. It is interesting to note how, in the incremental approach, the likelihood improves rapidly with increasing order until the proper order is reached. Thereafter only marginal improvement, which can be attributed to specialization on the training data, takes place.

4.4. Investigating the convergence of the FIT algorithm

As seen in the previous section, the FIT algorithm will not necessarily converge to the same values as the extended/ORED approach. Underlying training algorithms, such as the Baum-Welch algorithm, only guarantee convergence to a local optimum (Poritz, 1988). The FIT algorithm starts out with a different number of, and differently valued, initial parameters from that of the extended/ORED approach. It may therefore converge to a different local optimum. To investigate the quality of the FIT solution, carefully controlled experiments are needed. In particular, it is desirable to use data obtained from a hidden Markov source with known order and transition probabilities. Clearly, this is not possible with speech data, so synthetic data was used. This allows precise control over the Markov order, access to the underlying transition probabilities and control of the difficulty of the problem. For each synthetic experiment, two HMMs of a given order and complexity were generated. The extended/ORED and FIT algorithms were tasked with inferring the parameters of these models using data that was generated by them. Test data generated by the two models is then classified according to which model generated it. Access to the underlying HMMs allows testing against the actual generating models (not the FIT or ORED estimated models), and therefore allows an estimate of the optimal classification performance achievable. This is useful as it can identify specialization problems and determine expected performance bounds. The variety of conditions tested are summarized in Table III. The HMM structures used were generated by pruning links at random from higher-order ergodic HMMs and initializing the remainder in a random fashion. The means of the diagonal Gaussian pdf.s associated with each state were randomly positioned on a square two-dimensional plane. All the pdf.s shared a constant variance on both dimensions. The spacing between each pdf. and its closest neighbour was controlled as specified in Table III. For each of these conditions, 20 different models were investigated.

We first consider the compactness of models derived via the ORED and FIT algorithms as this indicates the computational complexity of the training procedure. In Table IV the excess number of non-zero transition probabilities (expressed as a

TABLE III. The four different generating HMM conditions evaluated. "σ Spacing" is the distance between each Gaussian centroid and its closest neighbour

Configuration number	1	2	3	4
Order (*R*)	2	3	4	2
States (*N*)	8	8	8	32
σ Spacing	1·5–3·0	1·5–3·0	1·5–3·0	1·0–2·0
Max links	656	5264	42128	34880
Ave links	134	398	1232	974
Sparseness (%)	20·4	7·6	2·9	2·8

TABLE IV. Excess transitions in ORED and FIT trained HMMs relative to that of the true underlying model. *R* is the order and *N* is the number of states

R,*N*	2,8	3,8	4,8	2,32
ORED (%)	109	453	1820	1203
FIT (%)	12	15	15	338

TABLE V. Error increase when classifying independent testing data with ORED, FIT and first-order trained HMMs, compared to that of the true underlying model

	Training set			Testing set		
R,*N*	ORED (%)	FIT (%)	HMM1 (%)	ORED (%)	FIT (%)	HMM1 (%)
2,8	19	32	112	40	41	120
3,8	6	34	146	79	52	151
4,8	−4	11	169	140	48	170
2,32	−7	22	237	212	115	238

percentage relative to the actual number of non-zero transition probabilities) resulting from these algorithms when compared to the true underlying model is shown. The sizes of the eight state models, which have fairly well separated Gaussian centroids, are well approximated by the FIT algorithm, irrespective of the sparseness factor. This, however, *does not hold for the extended/ORED approach, which rapidly escalates to computationally uneconomical sizes* (Table IV). We observe that the extended/ORED approach, although doing very well on the training data (even better than the true generating model), runs into serious trouble on the independent testing data, especially with the large, difficult models as shown in Table V. Correlating these results with that of Table IV indicates that the extended/ORED algorithm is likely to over-specialize. A McNemar test (Gillick & Cox, 1989) with a 95% confidence level shows that in the testing set only the ORED

TABLE VI. Comparison of 16-state ergodic HMMs trained via ORED and FIT. Accuracy measured on independent NIST'95 LID set. First-order accuracy is 82·1%

	Ratio FIT/ORED			Accuracy (NIST'95 LID)	
Order	CPU (%)	MEM (%)	Links (%)	ORED (%)	FIT (%)
2	94	69	70	87·2	89·7
3	7	13	5	89·7	97·4

vs. FIT results on the data from the (2,8) model, and the ORED vs. HMM1 results on the (2,32) HMM, do not differ significantly. The small number of parameters in the former case possibly forced the extended/ORED model to generalize. The latter is interesting as it indicates that overspecialization may rob a large higher-order HMM of its advantages over a first-order model; FIT models do not seem to be so susceptible to this.

5. Automatic language recognition experiments

We are currently investigating the use of high-order ergodic HMMs for automatic language recognition (ALR). In the following experiment we used roughly 100 minutes each of free English and Hindi speech (from the OGI-TS database) as training data. Silence sections in each recording were removed automatically, after which the energy was normalized. After pre-emphasis, LPC-cepstra and delta-cepstra were calculated at 16 ms intervals. This was followed by cepstral mean subtraction. Both languages were modelled with sixteen-state first, second and third-order ergodic HMMs. To provide robustness, each set of HMMs utilized a shared set of diagonal Gaussian pdf.s, thereby forcing the system to focus on phonotactics and not so much on acoustic-phonetics (Du Preez, 1993). Training followed both the ORED and FIT approaches.

Table VI summarizes the results. The memory ratios are based on the space required by both the Viterbi-paths matrix, as well as a sparse representation of the transition probabilities themselves. For the CPU calculation, only the transition probability component is considered. For models with many transitions and relatively few pdf.s, this dominates the CPU usage. Clearly the FIT approach results in more compact models. The classification accuracies confirm the results of the simulations of the previous section. However, due to the rather small test set (39 classifications—NIST'95 whole story specification), a McNemar test could only show the third-order FIT model to be better than the first-order HMM on a 90% significance level. No significant differences were detected between the ORED and FIT trained models.

6. Conclusion

The ORED algorithm presented in this paper allows any order HMM to be represented by an equivalent first-order model. This implies that all existing algorithms used for processing first-order HMMs can be directly applied to higher-order HMMs. This alone should encourage more researchers to experiment with the enhanced capabilities

of such models. Such first-order equivalents also simplify the interpretation of higher-order HMMs. The better control over allowable state sequences and improved duration modelling are clearly illustrated.

An efficient algorithm for training higher-order HMMs, based on the incremental training of HMMs of successive order (the FIT algorithm), is also given. It is shown to generalize better on previously unseen data. Its efficiency is verified on a language recognition task and demonstrates a practical training algorithm for higher-order HMMs.

The author wishes to thank the Oregon Graduate Institute for making their Multi-Language Telephone Speech (OGI-TS) database available. Etienne Barnard, Niko Brümmer and David Weber provided valuable help with their constructive comments made during the writing of this article. The Defence Research Development Council of South Africa and the Telkom-DataFusion Centre of Excellence is thanked for providing financial assistance.

References

Bahl, L. R., Jelinek, F. & Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5 no. 2, pp. 179–190.

Du Preez, J. A. (1993). *Discriminating between languages by means of Hidden Markov Models.* Internal research report, University of Stellenbosch, South Africa.

He, Y. (1988). Extended Viterbi algorithm for second order hidden Markov process. *Proceedings of the IEEE 9th International Conference on Pattern Recognition*, pp. 718–720. Rome, Italy.

Gillick, L. & Cox, S. J. (1989). Some statistical issues in the comparison of speech recognition algorithms. *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, pp. 532–535. Glasgow, U.K.

Gray, R. M. (1984). Vector quantization. *IEEE Acoustics Speech and Signal Processing Magazine*, vol. 1 no. 2, pp. 4–29.

Jelinek, F., Mercer, R. L. & Roukos, S. (1992). Principles of lexical language modelling for speech recognition. In *Advances in speech signal processing* (S. Furui and M. M. Sondhi, eds), Marcel Dekker, New York.

Kriouile, A., Mari, J.-F. & Haton, J.-P. (1990). Some improvements in speech recognition based on HMM. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 545–548. Albuquerque, U.S.A.

Lee, K. F. (1989). *Automatic Speech Recognition: The Development of the SPHINX System.* Kluwer Academic publishers.

Levinson, S. E. (1986). Continuously variable duration hidden Markov models for automatic speech recognition. *Computer Speech and Language*, vol. 1 no. 1, pp. 29–45.

Mari, J.-F. & Haton, J.-P. (1994). Automatic word recognition based on second-order hidden Markov models. In *ICSLP-94*, pp. 247–250.

Mari, J.-F., Fohr, D. & Junqua, J. C. (1996). A second-order HMM for high performance word and phoneme-based continuous speech recognition. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 435–438. Atlanta, U.S.A.

Mari, J.-F., Haton, J.-P. & Kriouile, A. (1997). Automatic word recognition based on second-order hidden Markov models. *IEEE Transactions on Speech and Audio Processing*, vol. 5 no. 1, pp. 22–25.

Poritz, A. B. (1988). Hidden Markov models: a guided tour. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7–12. New York, U.S.A.

(*Received 24 March 1997 and accepted for publication 8 December 1997*)

Appendix A: proof that ORED results in equivalent models

Section 3 shows that the normal *R*th-order model, and its first-order equivalent, apply the same free parameters in a different structure. The following proofs show that a second-order model, and its first-order version resulting from the ORED algorithm, are mathematically equivalent. The Corollary then extends this result to higher-order

models. In the following, the second-order HMM is M_2 and its equivalent first-order HMM is M_1. M_2 consists of N emitting states, augmented by an initial and a terminal null state.

Definition: Any two models M_1 and M_2 are defined as equivalent if $\mathrm{f}(\mathbf{X}_1^L|M_2)=\mathrm{f}(\mathbf{X}_1^L|M_1)$ for any arbitrary observation sequence $\mathbf{X}_1^L$. In other words, two models are only considered as equivalent if they yield the same likelihood, regardless of the specific observation sequence.

Theorem 1: Consider any sequence of states that may validly follow one other in a second-order HMM. Call it $Q_0^{L+1}=\{Q_0=0, Q_1, Q_2, \ldots, Q_L, Q_{L+1}=N+1\}$. For the first-order equivalent model, construct the following state sequence:

$$\begin{aligned} S_0^{L+1} &= \{S_0=Q_0, S_1 \\ &= (Q_0, Q_1), \ldots, S_{\ell-1}=(Q_{\ell-2}, Q_{\ell-1}), S_\ell=(Q_{\ell-1}, Q_\ell), \ldots, S_{L+1}=(Q_L,Q_{L+1})\} \end{aligned} \quad \text{(A.1)}$$

Then $Q_0^{L+1}\leftrightarrow S_0^{L+1}$ forms a one-to-one mapping of all valid state sequences in the second-order HMM to all valid state sequences in its first-order equivalent.

Proof: First consider the forward mapping $Q_0^{L+1}\leftrightarrow S_0^{L+1}$. Note that, from the algorithm of Section 3, the states in M_1 are specifically created to represent either the first state of M_2, or combinations of coupled states, making the constructed states valid for M_1. M_1 is constructed to allow transitions from state $S_{\ell-1}=(Q_{\ell-2}, Q_{\ell-1})$ to $S_\ell=(Q_{\ell-1}, Q_\ell)$ if it were legal in model M_2 to go from state $Q_{\ell-2}$ to $Q_{\ell-1}$ to Q_ℓ. Therefore the sequence of the constructed states is also valid. It is also clear from this construction that any specific Q_0^{L+1} maps to a unique S_0^{L+1}.

Now consider the backward mapping $Q_0^{L+1}\leftarrow S_0^{L+1}$. The uniqueness once again follows directly from the construction. From the algorithm of Section 3, there will only be a transition present between states $S_{\ell-1}=(Q_{\ell-2}, Q_{\ell-1})$ and $S_\ell=(Q_{\ell-1}, Q_\ell)$ if there were transitions from state $Q_{\ell-2}$ to $Q_{\ell-1}$ to Q_ℓ. Therefore, if S_0^{L+1} is a valid sequence, Q_0^{L+1} will also be so.

Theorem 2: The likelihood of an observation sequence $\mathbf{X}_1^L$ and a specific state sequence Q_0^{L+1} of the second-order HMM is equal to the likelihood of $\mathbf{X}_1^L$ and the associated unique state sequence S_0^{L+1} (defined in Theorem 1) of the equivalent first-order HMM, i.e. $f(\mathbf{X}_1^L, Q_0^{L+1}|M_2)=\mathrm{f}(\mathbf{X}_1^L, S_0^{L+1}|M_1)$.

Proof: Firstly, determine the likelihood using M_2. Using the definition of conditional probability, as well as the HMM assumptions 1 and 2, yields:

$$\begin{aligned} \mathrm{f}(\mathbf{X}_1^L, Q_{0+1}^L|M_2) &= \mathrm{P}(Q_{L+1}|\mathbf{X}_1^L, Q_0^L, M_2)\mathrm{f}(\mathbf{X}_1^L, Q_0^L|M) \\ &= \mathrm{P}(Q_{L+1}|Q_{L-1}^L, M_2)\mathrm{f}(\mathbf{x}L|\mathbf{X}_1^{L-1}, Q_0^L, M_2)\mathrm{f}(\mathbf{X}_1^{L-1}, Q_0^L|M_2) \\ &= a'_{Q_{L-1}Q_LQ_{L+1}}\mathrm{f}(\mathbf{x}_L|Q_L, M_2)\mathrm{f}(\mathbf{X}_1^{L-1}, Q_0^L|M_2). \end{aligned} \quad \text{(A.2)}$$

This last factor on the right hand side of (A.2) is of the same form as the left hand side. Recursive application of this expression then yields:

$$\begin{aligned} \mathrm{f}(\mathbf{X}_1^L, Q_0^{L+1}|M_2) = a'_{Q_{L-1}Q_LQ_{L+1}}\mathrm{f}(\mathbf{x}_\mathrm{L}|Q_\mathrm{L}, M_2)a'_{Q_{L-2}Q_{L-1}Q_L} \\ \cdots \mathrm{f}(\mathbf{x}_2|Q_2, M_2)a'_{Q_0Q_1Q_2}\mathrm{f}(\mathbf{x}_1|Q_1, M_2)a'_{Q_0Q_1}. \end{aligned} \quad \text{(A.3)}$$

Following the development of (A.3), the likelihood of M_1 can be shown to be

$$f(\mathbf{X}_1^L, S_0^{L+1}|M_1) = a_{S_0S_1}\mathrm{f}(\mathbf{x}_1|S_1, M_1)a_{S_1S_2}\mathrm{f}(\mathbf{x}_2|S_2, M_1) \\ \ldots a_{S_{L-1}S_L}\mathrm{f}(\mathbf{x}_L|S_L, M_1)a_{S_LS_{L+1}}. \tag{A.4}$$

Substitute (A.1) into (A.3) to obtain

$$\mathrm{f}(\mathbf{X}_1^L, S_0^{L+1}|M_1) = a_{Q_0(Q_0,Q_1)}\mathrm{f}(\mathbf{x}_1|S_1=(Q_0, Q_1), M_1)a_{(Q_0,Q_1)(Q_1,Q_2)}\mathrm{f}(\mathbf{x}_2|S_2=(Q_1, Q_2), M_1) \\ \cdots a_{(Q_{L-2},Q_{L-1})(Q_{L-1},Q_L)}\mathrm{f}(\mathbf{x}_L|S_L=(Q_{L-1}, Q_L), M_1)a_{(Q_{L-1},Q_L)(Q_L,Q_{L+1})}. \tag{A.5}$$

Now from step 3 of the ORED algorithm we have

$$a_{Q_0(Q_0,Q_1)} = a'_{Q_0Q_1} \\ a_{(Q_{\ell-2},Q_{\ell-1})(Q_{\ell-1},Q_\ell)} = a'_{Q_{\ell-2}Q_{\ell-1}Q_\ell} \qquad 2 \le \ell \le L+1, \tag{A.6}$$

and also

$$\mathrm{f}(\mathbf{x}_\ell|S_\ell=(Q_{\ell-1}, Q_\ell), M_1) = \mathrm{f}(\mathbf{x}_\ell|Q_\ell, M_2) \qquad \ell = 1 \cdots L. \tag{A.7}$$

Substituting (A.6) and (A.7) into (A.5) yields:

$$\begin{aligned}\mathrm{f}(\mathbf{X}_1^L, S_0^{L+1}|M_1) &= a'_{Q_0Q_1Q_0}\mathrm{f}(\mathbf{x}_1|Q_1, M_2)a'_{Q_0Q_1Q_2}\mathrm{f}(\mathbf{x}_2|Q_2, M_2) \\ &\quad \cdots a'_{Q_{L-2}Q_{L-1}Q_L}\mathrm{f}(\mathbf{x}_L|Q_L, M_2)a'_{Q_{L-1}Q_LQ_{L+1}} \\ &= \mathrm{f}(\mathbf{X}_1^L, Q_0^{L+1}|M)\end{aligned}$$

Theorem 3: The likelihood of attributing an observation sequence to a given second-order HMM is equal to being attributed to its transformed first-order version.

Proof: Theorem 2 state that $\mathrm{f}(\mathbf{X}_1^L, Q_0^{L+1}|M_2) = \mathrm{f}(\mathbf{X}_1^L, S_0^{L+1}|M_1)$, where the two state sequences correspond as is defined in Theorem 1. It is also known from Theorem 1 that for these two models there is a one-to-one mapping of such sequences. Therefore

$$\sum_{\forall S_0^{L+1}} \mathrm{f}(\mathbf{X}_1^L, S_0^{L+1}|M_1) = \sum_{\forall Q_0^{L+1}} \mathrm{f}(\mathbf{X}_1^L, Q_0^{L+1}|M_2) \tag{A.8}$$

The respective state sequences, over which is summed, are mutually exclusive and cover the whole sample space of such sequences (they form a partition). Therefore both sides of (A.8) reduce to marginal pdf.s giving the required result:

$$\mathrm{f}(\mathbf{X}_1^L|M_2) = \mathrm{f}(\mathbf{X}_1^L|M_)$$

Corollary: The likelihood of attributing an observation sequence to a given Rth-order HMM is equal to being attributed to its transformed first-order version.

Proof: An Rth-order model simply extends the memory of an $R-1$th-order model by one time step. Consider a second-order HMM, transformed to its first-order equivalent according to the given algorithm. Now replace each of the transition probabilities in the equivalent first-order model by the multiple transition probabilities of the second-order model with the same structure. More formally stated, let A_j be the

set of states that directly precedes state $S=j$. Replace the probability a_{jk} that is joining the states $S=j$ and $S=k$ by the multiple probabilities a_{ijk} where $i \in A_j$. The resulting model is the second-order equivalent of the third-order version of the original model. Using the algorithm of Section 3 this model can, of course, once again be transformed to an equivalent first-order model, thus yielding the first-order equivalent of a third-order model. Note that, although the base model is third-order, the transformation was still used to transform a second-order to first-order model. The proof for equivalence that was previously supplied therefore still applies. Repeated application of this process shows that a HMM of any order has a first-order equivalent.

Appendix B: basis of the FIT algorithm; transitions in a first-order HMM trained on second-order observations

In this section we investigate the behaviour of transition probabilities when training a first-order model (M_1) on data that is described by the second-order model (M_2) of similar structure. Consider the first-order probability in model M_2 of state $Q=j$, $j>0$ being followed by state $Q=k$ (without any consideration of earlier states). Transitions originating at the initial state are excluded since they are inherently of first-order. Let A_j represent the set of state that have a direct transition leading to state $Q=j$. Using the definition of total probability, conditional probability and the Bayes rule yields

$$\begin{aligned} \mathrm{P}(Q_\ell = k \,|\, Q_{\ell-1} = j, M_2) &= \sum_{i \in A_j} \mathrm{P}Q_\ell = k, Q_{\ell-2} = i \,|\, \mathrm{Q}_{\ell-1} = j, M_2), \quad j>0, \ell>1 \\ &= \sum_{i \in A_j} \mathrm{P}(Q_\ell = k \,|\, Q_{\ell-1} = j, Q_{\ell-2} = i, M_2) \mathrm{P}(Q_{\ell-2} = i \,|\, Q_{\ell-1} = j, M_2) \\ &= \sum_{i \in A_j} a'_{ijk} \mathrm{P}(Q_{\ell-1} = j \,|\, Q_{\ell-2} = i, M_2) \frac{\mathrm{P}(Q_{\ell-2} = i \,|\, M_2)}{\mathrm{P}(Q_{\ell-1} = j \,|\, M_2)}. \end{aligned} \tag{B.1}$$

Due to the $\mathrm{P}(Q_{\ell-2}=i|M_2)/\mathrm{P}(Q_{\ell-1}=j|M_2)$ factor, the required probability is a function of time. This suggests that the closest approximation to a second-order HMM using a first-order HMM of similar structure would make use of restricted time-varying transition probabilities. To make this time-dependence explicit, (B.1) can be rewritten as

$$\begin{aligned} a_{jk}(\ell) &= \sum_{i \in A_j} a'_{ijk} a_{ij}(\ell-1) \frac{P(Q_{\ell-2} = i \,|\, M_2)}{P(Q_{\ell-1} = j \,|\, M_2)}, \quad j>0, \ell>1. \\ &= \sum_{i \in A_j} a'_{ijk} w_{ij}(\ell-1) \quad \text{with} \quad w_{ij}(\ell-1) = a_{ij}(\ell-1) \frac{P(Q_{\ell-2} = i \,|\, M_2)}{P(Q_{\ell-1} = j \,|\, M_2)} \end{aligned} \tag{B.2}$$

If we assume that $Q=i$ is reachable from the initial state (no dead states), it implies that $\exists_\ell w_{ij}(\ell-1)>0$. From (12) it follows that $\exists_k a'_{ijk} \neq 0 \Rightarrow \exists_\ell a_{jk}(\ell) \neq 0$. In other words, if at least one of a set of second-order transition probabilities is non-zero, then the time-varying first-order approximation will be non-zero somewhere in time. First-order HMMs do not account for the time varying nature of (B.2), but will approximate

it with a single constant, let's say $a_{jk} = g(a_{jk}(\ell))$ where g() represents the unknown approximation. If $\exists_\ell a_{jk}(\ell) \neq 0$, any reasonable approximation $a_{jk} = g(a_{jk}(\ell))$ will result in $a_{jk} \neq 0$. Therefore, for any reasonable g() it follows that $\exists_i a'_{ijk} \neq 0 \Rightarrow a_{jk} \neq 0$. This leads us to conclude that if any in a set of second-order transition probabilities are greater than zero, any reasonable (constant) first-order approximation of it will also be greater than zero. This proves that trained first-order probabilities can be utilized to determine which sets of second-order transition probabilities are viable, thereby avoiding the training of redundant parameters.