ISSUES RELATED TO THE ESTIMATION OF TIME VARIANT HIDDEN MARKOV MODELS

Solomon Lerner and Baruch Mazor

GTE Laboratories Incorporated
40 Sylvan Road
Waltham, MA 02254

ABSTRACT

The Baum-Welch (BW) algorithm was originally developed to estimate the parameters of a time invariant hidden Markov model from a single training sequence. However, in speech applications, the BW algorithm is used to estimate *time varying* models from *multiple* training sequences. This use of the estimation algorithm, although very common, has not been thoroughly analyzed. In this work, we identify two of the several problems that we have found to be related to this use and describe novel solutions to alleviate them. Accordingly, we modified the basic estimation procedure and produced marked improvement in the fit of the trained models to the training data.

I. INTRODUCTION

An HMM in which there is an ordering of states such that transitions are allowed only from a low numbered state to a high numbered state is called a left-right (LR) or time varying model. Since the LR model mates the forward movement of time with a forward progression of states, it is an intuitive model for speech production. The topology commonly used, an example of which is shown in Fig. 1, is parameterized by JD, the maximum jump depth. JD characterizes the *maximum* number of states (excluding the present state) that can be jumped to from any particular state.

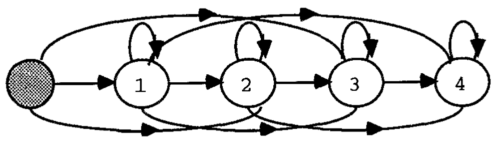


Fig. 1. A Left-Right HMM, JD=3.

With a JD of 1 only jumps from a state to itself or to its immediate neighbor are allowed. For an LR model, this implies that all states must be visited at least once. Alternatively, a JD equal to the number of states allows any state to skip to any other state and is the least constrained model. Most word based recognition systems use severely constrained models (JD=1). However, we feel that the richness and variability of speech can be modeled more accurately by a larger JD.

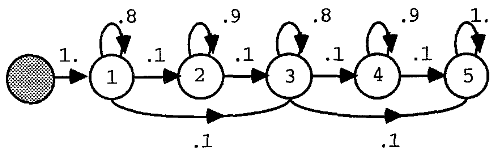
When JD is greater than one, a kind of transition that we will call a skip transition exists. A skip transition is a transition which causes the sequence to skip over one or more states. In time invariant models, there is a high probability of visiting a skipped state later in time; therefore, the parameters of such a state can be faithfully estimated from the sequence. However, the presence of skip transitions in LR models, means that some states may *never* be visited by a particular sequence. Thus, if we detect such a transition, the estimated parameters of the skipped state should not be updated based on that sequence. The capability to detect this condition and to act accordingly does not exist in current training algorithms.

During experiments, we have noticed the absence of this ability to cause a significant amount of estimation error in the parameters of these skipped states. In addition, we will show that the BW-based multi-sequence averaging algorithm has an additional adverse effect on the estimation of this kind of transition.

Another problem affecting the estimation results is the initialization procedure. Since the BW estimation procedure guarantees convergence to *local* maxima, the initialization algorithm may bias the estimation to converge to a sub-optimal maximum. Because of the increased complexity of the probability surface, this sensitivity to initial conditions is more pronounced for JD greater than one.

II. SKIP TRANSITIONS

Fig. 2 displays the SRC295 model. This model is basically the one used by Levinson, *et al.* [1], to illustrate the workings of the BW algorithm for LR models. In this paper, we will use it for the same purpose. We use a Monte-Carlo approach to generate training sequences. After training HMM's with these sequences, we compare the resulting models to the original SRC295 model. In this way, we can make a clear assessment on the correctness of the estimated models.



Symbol	1	2	3	4	5	6	7	8	9
State 1	.7	.3	.0	.0	.0	.0	.0	.0	.0
State 2	.0	.0	.8	.2	.0	.0	.0	.0	.0
State 3	.0	.0	.0	.0	1.0	.0	.0	.0	.0
State 4	.0	.0	.0	.0	.0	.2	.8	.0	.0
State 5	.0	.0	.0	.0	.0	.0	.0	.3	.7

Fig. 2. The SRC295 HMM.

Notice that in the SRC295 model, states 2 and 4 can be skipped by transitions originating at states 1 and 3, respectively. A particular training sequence may have been generated by a state sequence that utilized such a *skip transition.* If this is the case, the sequence never visited one of these skipped states and should not be used to estimate the parameters associated with the state. Trivially, this effect is a direct result of having a JD greater than one.

The other issue affecting model estimation is that longer sequences dominate the estimation when the common multi-sequence/BW averaging algorithm [1] is used. To demonstrate this bias for the transitional parameters, we can write their estimation equation [1] using the following definitions:

$$N_k = \frac{1}{P_k} \sum_{t=1}^{T-1} a_t^k(i)\, a_{ij}\, b_i(O_{t+1}^k)\, b_{t+1}^k(j) \qquad (1a)$$

$$D_k = \frac{1}{P_k} \sum_{t=1}^{T-1} a_t^k(i)\, b_t^k(i) \,/\, c_t \qquad (1b)$$

where k is the sequence index, t is the time index, a and b are the forward/backward values, P_k is the probability of generating sequence k, c_t is a scaling parameter, and a_{ij} and b_i are previously estimated HMM parameters. The new estimate of a_{ij} is just the sum over k of N_k divided by the sum over k of D_k. From a cursory glance at Eqn. 1a and 1b, we may notice that the time summations are of the same order of magnitude for each sequence k. However, P_k has a very strong dependence on length- the longer the sequence the smaller the probability of generating it. Thus, the values of N_k and D_k are strongly dependent on the longer sequences.

To demonstrate the effect of length experimentally, we generated 500 sequences from the SRC295 model. For each sequence length L, LN(L), the average of logN (for i, j = 1) over all sequences of length L, was computed. Fig. 3 is the resulting graph of LN(L) versus L. Note the linear relationship between LN(L) and L which means an exponential relationship in the linear domain. Since a similar relationship exists between D and L, it is clear that the longer sequences dominate the estimation of the transitional parameters!

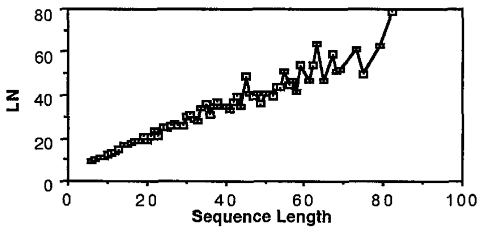


Fig. 3. LN(L) vs L.

When estimating models such as the SRC295, we have seen many problems caused by this length weighting. Most noticeably, we have observed cases where the estimated model reflects only the longest sequences. Since these sequences are most probably associated with a state sequence without skipped states, the estimated probabilities of the skip transitions are usually close to zero. Thus, the effective JD of the resulting models is 1! For example, when we used sequences generated from the SRC295 model in 25 training runs, each with its own random initial conditions and with JD initialized to 3, *all skip transitions* from *all of the 25 resulting models* had probabilities smaller than 10^{-20}. However, skip transitions due to the initial state probabilities (π vector) were consistently observed. Additionally, the final fit of the model to the training sequences was very poor.

In order to alleviate these effects, it is important to decide whether a particular sequence resulted from a skip state transition based on the observed sequence and a prior model estimate. Before we discuss the proposed solution, we introduce the quantity PV(i), which is the probability of visiting state i. Given an HMM, PV(i) is computed recursively as follows:

$$PV(i) = \pi_i + \sum_{j=1}^{i-1} PV(j)\,\frac{a_{ji}}{1-a_{jj}} \qquad (2)$$

In our solution, called SKIP-TRAIN, we compute PV(i) from parameters that were reestimated using *only* the given sequence. If PV(i) is greater than some threshold, we assume that state i was visited during generation of the sequence and we use this sequence in the estimation of the state parameters. Conversely, if PV(i) is below the threshold, then this sequence is not considered in the estimation of the parameters of state i.

III. INITIALIZATION

In our first experiments with the SRC295 model, we used the following initialization algorithm which is similar to those commonly described in the literature:

Output symbol density (B matrix)

$b_{jk} = 1/M; \qquad j \in [1, N], k \in [1, M]$

Initial state probabilities (π vector)

$tmp_i = 1/JD^* + 0.1{*}RND(); \qquad i \in [1, JD^*]$

$$\pi_i = tmp_i / \sum_{j=1}^{JD^*} tmp_i \qquad \text{(normalize)}$$

Transitional probabilities (A matrix)

$tmp_{ij} = 1/(JD+1) + 0.1{*}RND(); \qquad i \in [1, N], j \in [i, i+JD]$[1]

$$a_{ij} = tmp_{ij} / \sum_{j=i}^{i+JD} tmp_{ij} \qquad \text{(normalize)}$$

Here, the RND function returns a random number uniformly distributed between -1 and +1.

One difference between our algorithm and those described in the literature is that we allow more than one starting state; a look at the above initialization algorithm will show, that there are a total of JD* states that can be used as starting states. We believe that this extra freedom in the model can compensate for inaccurate end-point detection. Furthermore, it allows different pronunciations of the first phoneme to be represented by different states. Although not mandatory, we will henceforth assume that JD* = JD.

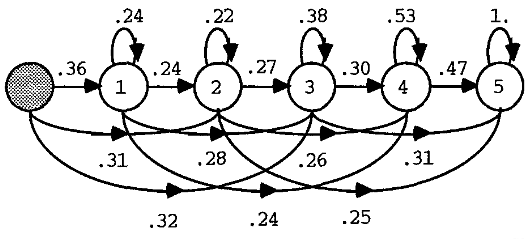
We propose two modifications for the initialization step. In the first modification, we use the training data to initialize the B matrix. In this procedure, called AVGINIT, all the training sequences are divided into N equal time segments corresponding to the N states. For each state, the tokens that were observed during its corresponding time segment are accumulated and the probability of observing the symbol is computed from the resulting average. Also, in order to minimize possible catastrophic effects we set the minimum observable probability to 0.0001.

The second modification is related to deleted states- a phenomenon which occurs frequently as JD increases. A deleted state is a state which has a very small estimated probability of being visited. This phenomenon was especially prevalent among the first two states and occurred more frequently when the SKIP-TRAIN estimation algorithm was used.

In Table 1, we list the probability of visiting each state for several different HMM's. Notice the relatively low probability associated with visiting the first state of the HMM of Fig. 4 which was initialized using the RAND algorithm alone. The range of this effect (i.e. the number of initial states with smaller PV's) is directly related to the value of JD.

State	PV Fig. 4	PV Fig. 5	PV Fig. 6	PV Fig. 7	PV SRC295
1	36%	0%	53%	100%	100%
2	43%.	0%	56%	45%	50%
3	60%	100%	64%	100%	100%
4	56%	45%	65%	66%	50%
5	100%	100%	100%	100%	100%

Table 1. The probability of visiting states for several different HMM's.



Symbol	1	2	3	4	5	6	7	8	9
State 1	.11	.11	.11	.11	.11	.11	.11	.11	.11
State 2	.11	.11	.11	.11	.11	.11	.11	.11	.11
State 3	.11	.11	.11	.11	.11	.11	.11	.11	.11
State 4	.11	.11	.11	.11	.11	.11	.11	.11	.11
State 5	.11	.11	.11	.11	.11	.11	.11	.11	.11

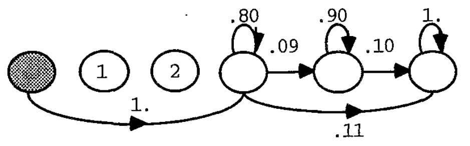
Fig. 4. Initial HMM computed using RAND routine.

This effect biases the estimate towards deleting the initial states from the HMM. For example, Fig. 5 is the result of a training run using the HMM

[1] Actually, j ranges from i to min(i+JD, N) and (min(i+JD, N)-i+1) should replace (JD+1).

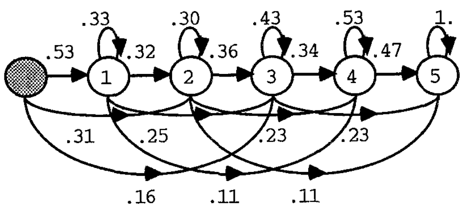
of Fig. 4 for initialization. A glance at Table 1 shows that the probabilities of visiting the first two states is zero; i.e. these states have been deleted!

Based on this observation, we introduced a second modification to the initialization procedure called WINDINIT. After the random transitional parameters were computed (using the RAND algorithm), we linearly weighted the non self-transition parameters in such a way that transitions to nearer states were favored. We then renormalized the probability parameters. Fig. 6 shows the HMM of Fig. 4 after both the AVGINIT and WINDINIT routines have been used. From the fourth column of Table 1, we can see that the probabilities of visitation are better equalized in the initial model by this simple modification.



Symbol	1	2	3	4	5	6	7	8	9
State 1	–	–	–	–	–	–	–	–	–
State 2	–	–	–	–	–	–	–	–	–
State 3	.69	.31	.00	.00	.00	.00	.00	.00	.00
State 4	.00	.00	.83	.17	.00	.00	.00	.00	.00
State 5	.00	.00	.00	.00	.33	.06	.22	.13	.26

Fig. 5. Trained HMM initialized with RAND routine.



Symbol	1	2	3	4	5	6	7	8	9
State 1	.49	.23	.12	.03	.09	.00	.01	.00	.00
State 2	.18	.10	.27	.05	.24	.02	.08	.02	.02
State 3	.09	.03	.14	.05	.29	.06	.21	.07	.07
State 4	.00	.00	.08	.01	.22	.05	.32	.10	.19
State 5	.00	.00	.01	.00	.09	.06	.13F	.21	.48

Fig. 6. Initial HMM computed using AVGINIT and WINDINIT routines.

Likewise, Fig. 7 and the fifth column of Table 1 show the training results when the HMM of Fig. 6 is used as the initial guess. Notice how close the results are to the real SRC295 model shown in the last column. Additionally, by examining the estimated model, we can see that all the other model parameters are also in close agreement with the SRC295 model.

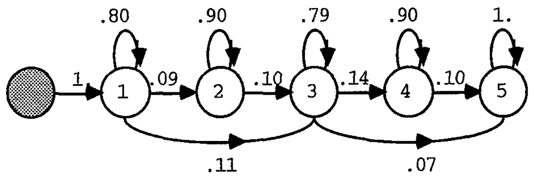
IV. EXPERIMENTS

In this section, we will present results of several large scale experiments conducted to illustrate the effects of the problems discussed in the previous sections and how our solutions have resolved them. The goal is to show that the improvements result in more accurate modeling of training data; i.e. we will not be concerned here with how these improvements affect speech recognition. However, we feel it safe to assume that improvements in the accuracy of these models will also improve recognition performance.

Results of HMM experiments are usually difficult to quantify because of the following problems:

1. The BW algorithm is guaranteed to converge only to *local* maxima. Thus, random model initializations can affect the final model and some form of average performance over many runs needs to be used.

2. When working with real world data, it is difficult to set performance criteria since the underlying HMM is not known.



Symbol	1	2	3	4	5	6	7	8	9
State 1	.69	.31	.00	.00	.00	.00	.00	.00	.00
State 2	.00	.00	.83	.17	.00	.00	.00	.00	.00
State 3	.00	.00	.00	.00	1.00	.00	.00	.00	.00
State 4	.00	.00	.00	.00	.00	.20	.80	.00	.00
State 5	.00	.00	.00	.00	.00	.00	.00	.33	.67

Fig. 7. Trained HMM initialized with AVGINIT and WINDINIT routines.

3. The stopping criteria for the reestimation process must be carefully set to ensure convergence.

We use the following techniques to mitigate these effects:

1. All reported results are based on 100 runs, each of which utilizes a unique initial model. Additionally, whenever comparing the performance of two algorithms which have the same model parameters (i.e. N, M, and JD), the same 100 initial conditions are used for both algorithms.

2. Since we are primarily interested in modeling performance, we used Monte-Carlo simulations to generate 50 training sequences from *known* HMM's. These same 50 sequences were used for testing purposes. Thus, the HMM is scored only on its ability to match the training data.

3. On each of the 100 runs, we iterated the BW training algorithm a constant 30 times.

As a performance measure, we will use the average log-probability of generating the training sequences given a model. We will call this measure LP(M) where M is the model being tested. If $P_k(M)$ is the probability of generating sequence k from model M, then LP(M) is the average of $\log_{10}P_k(M)$ over the fifty sequences. In order to provide a baseline for performance, we computed LP(SRC295) which was found to be -25.79. *All our results will be relative to this baseline.*

Skip Transitions

The performance of the BW algorithm without the SKIP-TRAIN and initialization improvements was evaluated using the performance figure $\overline{LP}$. This was just the average of LP(M) over all 100 final models. Table 2 lists the results of this evaluation for different starting values of JD.

In this table, the column entitled "# DS HMM's" indicates the number of trained HMM's (out of 100) which contain at least one deleted state. In these experiments, we will say that a state is deleted from an estimated model if the probability of visiting this state is less than 0.1. Observe that a significant portion of trained models contain deleted states. We will discuss the cause of these deleted state models later.

Initial JD	$\overline{LP}$	Best LP(M)	Worst LP(M)	# DS HMM's
1	-94.2	-94.1	-94.2	N/A
2	-117.5	-114.1	-120.9	15
3	-112.6	-109.6	-132.2	86
4	-124.3	-109.6	-132.1	59

Table 2. Performance results using RAND algorithm only.
Results are in dB relative to baseline performance.

As discussed in Section II, the SKIP-TRAIN algorithm uses the estimated probability of visiting a state in order to determine whether the state average should be updated from a particular sequence. In our experiments, the probability threshold was set to 70%. We applied this threshold beginning at the second iteration. Table 3 shows the results when the SKIP-TRAIN algorithm was used. Aside from some slightly worse results for JD = 4, which will be explained later, the SKIP-TRAIN algorithm is seen to produce a superior fit of the trained model to the data.

Initial JD	$\bar{LP}$	Best LP(M)	Worst LP(M)	# DS HMM's
1	-94.2	-94.1	-94.2	N/A
2	-75.7	-71.9	-116.0	87
3	-81.3	-71.2	-131.7	4
4	-125.4	-71.2	-149.6	87

Table 3 Performance results using RAND and SKIP-TRAIN algorithms.

Initialization

The results of Table 3 indicate that there are still several problems affecting the estimation procedure. The most obvious indication is the large numbers of trained HMM's with deleted states. These models generally perform worse than the others and are a symptom of an underlying problem that is causing underutilization of the model. The second indication of a problem is that all the performance figures displayed so far, *including the best ones*, are very far from the baseline figure. Therefore, an attempt to search the parameter space by running more experiments with different initial conditions is pointless.

As previously described, we introduced the WINDINIT and the AVGINIT routines to deal with these two problems. An example of the power of these routines is shown in Table 4 which lists the results of experiments in which the WINDINIT and AVGINIT routines were used for initialization. In this table, *all 100 runs* for *all values* of JD greater than 1 resulted with performance equal to the baseline log-probability. Furthermore, there was virtually *an exact match* between *all* trained models and the SRC295 model!

Another interesting point is that without the use of the SKIP-TRAIN algorithm, the benefit from these initialization algorithms was not as significant. For example, for the JD=3 case, the relative performance dropped to -72 dB when WINDINIT and AVGINIT were used without the SKIP-TRAIN algorithm. Thus we conclude that both the initialization algorithms and the SKIP-TRAIN algorithm are necessary for the performance of Table 4.

Initial JD	$\bar{LP}$	Best LP(M)	Worst LP(M)	# DS HMM's
1	-94.2	-94.1	-94.2	N/A
2	-0.1	-0.1	-0.1	0
3	-0.1	-0.1	-0.1	0
4	-0.1	-0.1	-0.1	0

Table 4. Performance using WINDINIT, AVGINIT, and SKIP-TRAIN.

What JD to Use?

From Table 3, we notice the anomalous result that increasing JD to 4 had a detrimental effect on performance. On the surface, this is surprising since a large value of JD increases the degrees of freedom in the system. On the other hand, it is evident that larger values of JD cause the probability surface to become more complicated which in turn increases the possibility of falling into a local maximum. Thus, a pertinent question is whether it is better to overestimate or to underestimate JD when the actual value is not known.

We have observed that the major factor limiting $\bar{LP}$ when overestimating JD, is the underutilization of the first and second states. This manifests itself both in deleted state models and in models whose first and second states have very similar output distributions ("identical" states). The introduction of the AVGINIT and WINDINIT routines seems to solve both problems in the SRC295 model. In addition, we can always detect deleted and "identical" states and discard any training runs in which they are observed. Since detection is a simple procedure, we believe that it may be used to complement the AVGINIT and WINDINIT routines when training more complicated models such as in speech modeling.

When the underutilization problems were solved for the SRC295 model, the case for using a large value of JD becomes strong. This is evident in Table 4 where *no penalty was observed for increasing JD* and conversely, a large penalty was observed for JD=1. Thus, we conclude that as long as the deleted and "identical" state problems are dealt with, it is better to overestimate JD.

V. DISCUSSION

There are two other ways to conceptualize the SKIP-TRAIN algorithm. Underlying HMM modeling is the Markovian assumption that the output probabilities only depend on the current state. Since this is clearly not accurate for speech, several attempts have been made to relax this assumption somewhat (see [3] for example). The SKIP-TRAIN algorithm implicitly relaxes this assumption since the current state is dependent on the branches taken to reach it and is therefore dependent on the previous output probabilities.

Additionally, the SKIP-TRAIN algorithm can be visualized as a clustering procedure. If all the different state sequences of the HMM are enumerated and regarded as different JD = 1 models, then the training problem is divided into two steps. First, decide which state sequence (cluster) each input belongs to. Second, using all inputs assigned to a particular state sequence, find a better centroid (i.e. train). In this description, SKIP-TRAIN is just the algorithm that assigns an input to a cluster. Of course, since a state can be shared by more than one cluster, there are constraints on this procedure that are not present in general clustering algorithms.

As a last comment, we wish to point out that this algorithm has similarities with the Viterbi segmentation (VS) algorithm [4] which uses a Viterbi decision rule to determine the state sequence. We can describe the relationships among these algorithms as follows: The BW algorithm updates *all* possible state sequences, the SKIP-TRAIN algorithm updates *some* of the most likely state sequences and the VS algorithm updates only *the most likely* state sequence. In preliminary tests, we have found that the performance of the SKIP-TRAIN and the VS algorithm are comparable. The only significant difference in performance that we have found is that in very few cases, the VS algorithm converged to a suboptimal solution when initialized using AVGINIT and WINDINIT. This is slightly inferior to the performance of the SKIP-TRAIN algorithm which converged to the optimal HMM in all cases. We surmise therefore, that the hard decisions made by the VS algorithm may sometimes cause a suboptimal decision by converging to the wrong maximum.

VI. CONCLUSION

We have described new algorithms for use in the estimation of LR hidden Markov models with skip transitions. The SKIP-TRAIN algorithm ensures that only the parameters of states with a high probability of having been visited by a sequence are updated. The WINDINIT and AVGINIT algorithms counteract the bias towards underutilizing the first several states. Via Monte-Carlo simulations and theoretical arguments, we have shown that these algorithms improved the modeling of time varying, multi-sequence data. Although not reported here, we confirmed our conclusions with a much more complex model- one with 7 states and a JD of 4. We feel that these results are important for speech modeling since large values of JD are probably necessary to model the different variations found in spoken language. In the future, we will explore the potential of this improved modeling in speech recognition.

REFERENCES

[1] Levinson, S., L. Rabiner, and M. Sondhi, "An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition," *BSTJ*, 62, 4, pp. 1035-1075, Apr. 1983.

[2] Rabiner, L., S. Levinson, and M. Sondhi, "On the Application of Vector Quantization and Hidden Markov Models to Speaker-Independent, Isolated Word Recognition," *BSTJ*, 62, 4, pp. 1075-1105, Apr. 1983.

[3] Wellekens, C., "Explicit Time Correlation in Hidden Markov Models for Speech Recognition," Proc. ICASSP-87, pp. 10.7.1-3.

[4] Rabiner, L., B. Juang, S. Levinson, and M. Sondhi, "Recognition of Isolated Digits Using Hidden Markov Models with Continuous Mixture Densities," *ATTTJ*, 64, 6, pp. 1211-1234, July 1985.