On the Complexity of Explicit Duration HMMs

Carl Mitchell, Mary Harper, and Leah Jamieson

Abstract— **We introduce a new recursion that reduces the complexity of training a semi-Markov model with continuous output distributions. We show that the cost of training is proportional to $M^2 + D$, compared to M^2D with the standard recursion, where M is the observation vector length and D is the maximum allowed duration.**

I. Introduction

In the course of our work on parallelizing HMM (hidden Markov model) algorithms [1], we have discovered a new recursion that reduces the number of vector operations in a continuous output, explicit duration HMM. Following a definition of terms in section II, we develop the result in three stages. In section III, we demonstrate that the complexity of the forward, backward, and Viterbi algorithms is $O(NKT(M^2 + D))$ using known recursions, where T is the number of observations, N is the number of states, K is the average number of predecessors across all states, and M is the observation vector length. In section IV, we show that the complexity of re-estimation is $O(NKTM^2D)$ using a recursion given by Levinson [2]. This demonstrates that the cost of explicit duration modeling is proportional to D, and not D^2 as is often quoted in the literature [3], [4]. Finally, we introduce a new recursion in section V that lowers the complexity of re-estimation (and therefore training) from $O(NKTM^2D)$ to $O(NKT(M^2 + D))$ for continuous HMMs. This improved complexity is achieved by delaying vector operations until D scalar weights have been accumulated.

II. Terminology

We use a semi-Markov model [5], [6] where the number of observations drawn from the same density (i.e., duration) is modeled explicitly. We assume that the observations $\{\boldsymbol{O}_1, .., \boldsymbol{O}_T\}$ are normally distributed, where boldface type indicates that the symbol refers to a vector. The output symbols are emitted during the transitions between states (i.e., a Mealy model) as shown in figure 1. Let K be defined as the average number of transitions to a state, where a transition exists from state i to state j if the transition probability, $a_{i,j}$, is greater than zero. There are NK transitions in the model:

$$\sum_{j=1}^{N} \sum_{\substack{i=1 \\ \ni a_{i,j} > 0}}^{N} 1 = NK$$

For a fully connected model, $K = N$. Arguments similar to those in this paper can be used for the case where the observations are instead produced from the states (i.e., a Moore model) or the output distributions are not Gaussian. Definitions for terms used in this paper are shown in table I. Let $d_{i,j}(\tau)$ be the probability mass function for the duration of the transition from state i to state j. We assume that the maximum duration is D. A duration of zero (i.e., a null transition) is permitted as long as a closed loop of null transitions does not occur. The equations given in this paper can be modified slightly to allow for zero duration.

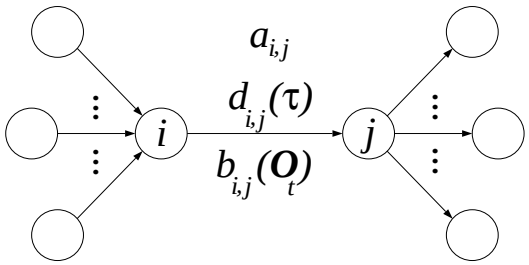

Fig. 1. The observations are emitted on the transitions.

III. Complexity of the Forward, Backward, and Viterbi Algorithms

Training a hidden Markov model consists of iteratively improving an estimate of λ, the set of model parameters.[1] An improved estimate can be conveniently and efficiently expressed in terms of the forward and backward probabilities.

Define $\alpha_t(j)$, the forward probability, as follows:

$$\alpha_t(j) = Pr(\boldsymbol{O}_1, \boldsymbol{O}_2, .., \boldsymbol{O}_t, \ state\ j\ ends\ at\ time\ t|\lambda)$$

$$\alpha_0(j) = \begin{cases} 1 & \text{if } j \text{ is the initial state} \\ 0 & \text{otherwise} \end{cases}$$

For $t = 1, .., T, \quad j = 1, .., N$:

$$\alpha_t(j) = \sum_{\substack{i=1 \\ \ni a_{i,j} > 0}}^{N} \sum_{\tau=1}^{\min(D,t)} \alpha_{t-\tau}(i) a_{i,j} d_{i,j}(\tau) \underbrace{\prod_{p=1}^{\tau} b_{i,j}(\boldsymbol{O}_{t-p+1})}_{U_{t,i,j}(\tau)}$$

Evaluating $b_{i,j}(\boldsymbol{O}_t)$ requires $O(M^2)$ operations since we assume that all M^2 elements of covariance matrix $\boldsymbol{\Sigma}_{i,j}$ could be non-zero.[2] The product of output probabilities can be

C. Mitchell, M. Harper, and L. Jamieson are with the School of Electrical Engineering, Purdue University, West Lafayette, IN 47907-1285. Email: {cdm,harper,lhj}@ecn.purdue.edu

[1] See [3] for an excellent introduction to hidden Markov models for speech recognition.

[2] If $\Sigma_{i,j}$ is assumed diagonal, then every occurrence of M^2 in this analysis can be replaced by M.

TABLE I
THE ELEMENTS OF A HIDDEN MARKOV MODEL.

Term	Definition
N	Number of states in the HMM (typically ~ 2500).
K	Average number of predecessors for a state, which is the total number of transitions divided by the total number of states, N (typically ~ 2).
$a_{i,j}$	Probability that the next state will be j, given that the current state is i.
T	Number of observations in an utterance (typically ~ 300 per sentence, or 1,000,000 if all training sentences are concatenated).
$\boldsymbol{O}_1^T$	Observation sequence, $\{\boldsymbol{O}_1, ..., \boldsymbol{O}_T\}$, where observation $\boldsymbol{O}_t$ is a vector.
M	Dimension of the observation vector (typically ~ 25).
$b_{i,j}(\boldsymbol{O}_t)$	Probability that $\boldsymbol{O}_t$ will be emitted during the transition from state i to state j. We assume $b_{i,j}(\boldsymbol{O}_t)$ is Gaussian with mean $\boldsymbol{\mu}_{i,j}$ and covariance matrix $\boldsymbol{\Sigma}_{i,j}$.
D	Maximum number of observations for a single transition (typically ~ 25).
$d_{i,j}(\tau)$	The probability that τ output vectors will be emitted when the process switches from state i to state j.
λ	$(\{a_{i,j}\}, \{d_{i,j}(\tau)\}, \{\boldsymbol{\mu}_{i,j}\}, \{\boldsymbol{\Sigma}_{i,j}\})$, the complete set of model parameters.

calculated recursively [2]:

$$U_{t,i,j}(\tau) = \prod_{p=1}^{\tau} b_{i,j}(\boldsymbol{O}_{t-p+1}) = b_{i,j}(\boldsymbol{O}_{t-\tau+1}) U_{t,i,j}(\tau-1)$$
$$\text{for } \tau = 1, .., \min(D,t), \text{ where } U_{t,i,j}(0) = 1 \quad (1)$$

When summing over t, there are $\min(D,t)-1$ output probabilities needed for $U_{t,i,j}(\tau),\ \tau = 1, .., \min(D,t)$ that were calculated for $U_{t-1,i,j}(\tau),\ \tau = 1, .., \min(D,t-1)$. By saving these output probabilities, the complexity of (1) can be reduced from $O(M^2 D)$ to $O(M^2 + D)$. For each t, a column of the forward trellis can be efficiently updated as follows:

1. Calculate $b_{i,j}(\boldsymbol{O}_t)$ for all combinations of i and j in $O(NKM^2)$ operations.
2. Using saved values of $b_{i,j}(\boldsymbol{O}_s)$, $s = t-\min(D,t)+1, .., t$, calculate $U_{t,i,j}(\tau),\ \tau = 1, .., \min(D,t)$ for all combinations of i and j in $O(NKD)$ operations.
3. With the $U_{t,i,j}(\tau)$'s given, find $\alpha_t(j)$ for all j in $O(NKD)$ operations.

These steps are repeated for each t, leading to $O(NKT(M^2 + D))$ complexity for the forward algorithm.

Define $\beta_t(i)$, the backward probability, as follows:

$$\beta_t(i) = Pr(\boldsymbol{O}_{t+1}, \boldsymbol{O}_{t+2}, .., \boldsymbol{O}_T | \textit{ state } i \textit{ ends at time } t,\ \lambda)$$

$$\beta_T(i) = \begin{cases} 1 & \text{if } i \text{ is the final state} \\ 0 & \text{otherwise} \end{cases}$$

For $t = T-1, .., 0, \quad i = 1, .., N$:

$$\beta_t(i) = \sum_{\substack{j=1 \\ \ni a_{i,j} > 0}}^{N} \sum_{\tau=1}^{\min(D,T-t)} a_{i,j} d_{i,j}(\tau) \beta_{t+\tau}(j) \prod_{p=1}^{\tau} b_{i,j}(\boldsymbol{O}_{t+p})$$

The complexity of calculating the backward trellis is also $O(NKT(M^2 + D))$, which can be seen from the similarity between the forward and backward algorithms.

The Viterbi algorithm, which is used to recognize the most probable state path, is identical to the forward algorithm except that the maximum of all predecessors is taken rather than the sum. Thus the complexity of recognition is also $O(NKT(M^2 + D))$.

During training, the forward and backward probabilities are combined by the Baum-Welch re-estimation algorithm. The re-estimation algorithm is similar to the forward and backward algorithms, except that an additional sum is taken from 1 to D during re-estimation of the output parameters. Fortunately there are recursions available so that the cost of training is proportional to D.

IV. Complexity of Re-Estimation Using Standard Recursions

Calculating the output counts using the re-estimation algorithm is the most computationally demanding part of training. We will show that the complexity of this task is linear in the maximum allowed duration, using a recursion given by Levinson [2].

An improved estimate of the mean for transition $i \rightarrow j$ can be found in terms of the forward and backward prob-

abilities:

$$\hat{\boldsymbol{\mu}}_{i,j} = \frac{\hat{\boldsymbol{\mu}}_{i,j}^{num}}{\hat{\mu}_{i,j}^{den}} = \frac{\sum_{t=1}^{T}\sum_{\tau=1}^{\min(D,t)} P_{t,i,j}(\tau)\sum_{s=1}^{\tau}\boldsymbol{O}_{t-s+1}}{\sum_{t=1}^{T}\sum_{\tau=1}^{\min(D,t)} P_{t,i,j}(\tau)} \quad (2)$$

$$\text{where } P_{t,i,j}(\tau) = \alpha_{t-\tau}(i)a_{i,j}d_{i,j}(\tau)\beta_t(j)\prod_{p=1}^{\tau} b_{i,j}(\boldsymbol{O}_{t-p+1})$$

$$= \alpha_{t-\tau}(i)a_{i,j}d_{i,j}(\tau)\beta_t(j)U_{t,i,j}(\tau)$$

The term $P_{t,i,j}(\tau)$ is the joint probability that the process generates $\boldsymbol{O}_1$ through $\boldsymbol{O}_T$, leaves state i at time $t-\tau$, and arrives in the next state, j, at time t, given the model λ. Summing $P_{t,i,j}(\tau)$ over t yields the probability of all state sequences that produce the observation sequence and for which transition $i \rightarrow j$ occurs with duration τ.

Consider the complexity of the numerator, $\hat{\boldsymbol{\mu}}_{i,j}^{num}$, in (2):

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \sum_{t=1}^{T}\sum_{\tau=1}^{\min(D,t)} \alpha_{t-\tau}(i)a_{i,j}d_{i,j}(\tau)\beta_t(j)U_{t,i,j}(\tau)\boldsymbol{V}_t(\tau) \quad (3)$$

$$\text{where } \boldsymbol{V}_t(\tau) = \sum_{s=1}^{\tau}\boldsymbol{O}_{t-s+1}$$

The $\boldsymbol{V}_t(\tau)$ terms can be efficiently precalculated using the following recursion for each t:

$$\boldsymbol{V}_t(\tau) = \sum_{s=1}^{\tau}\boldsymbol{O}_{t-s+1} = \boldsymbol{O}_{t-\tau+1} + \boldsymbol{V}_t(\tau-1) \quad (4)$$

$$\text{for } \tau = 1,..,\min(D,t), \text{ where } \boldsymbol{V}_t(0) = \boldsymbol{0}$$

Precalculating the observation sums using (4) requires $O(TMD)$ operations. There is a similar term, $\boldsymbol{V}_t^{\Sigma}(\tau)$, used for estimating the covariance matrix ($\boldsymbol{O}_t$ is replaced by $\boldsymbol{O}_t\boldsymbol{O}_t^*$, where $*$ denotes transpose) that requires $O(TM^2D)$ operations. After the $\boldsymbol{V}_t(\tau)$'s have been saved, the means can be updated with the following steps for each i, j, and t:

1. Use (1) to recursively calculate the partial products of output probabilities, $U_{t,i,j}(\tau), \tau = 1,..,\min(D,t)$, in $O(M^2+D)$ operations.
2. Calculate the $P_{t,i,j}(\tau)$ terms in $O(D)$ operations.

$$P_{t,i,j}(\tau) = \alpha_{t-\tau}(i)a_{i,j}d_{i,j}(\tau)\beta_t(j)U_{t,i,j}(\tau)$$

$$\text{for } \tau = 1,..,\min(D,t)$$

3. Update the mean in $O(MD)$ operations.

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \hat{\boldsymbol{\mu}}_{i,j}^{num} + \sum_{\tau=1}^{\min(D,t)} P_{t,i,j}(\tau)\boldsymbol{V}_t(\tau)$$

$$\hat{\mu}_{i,j}^{den} = \hat{\mu}_{i,j}^{den} + \sum_{\tau=1}^{\min(D,t)} P_{t,i,j}(\tau)$$

For estimating the covariance matrix, there is a calculation that is similar to step 3, except that the vector sum is replaced by an $M \times M$ matrix sum, making the complexity $O(M^2D)$. These steps are repeated for all combinations of i, j, and t, so that re-estimating the output parameters has $O(NKTM^2D)$ complexity.[3]

V. Improved Complexity of Training Using A New Recursion

Again consider the numerator of the mean estimate.

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \sum_{t=1}^{T}\sum_{\tau=1}^{\min(D,t)} P_{t,i,j}(\tau)\sum_{s=1}^{\tau}\boldsymbol{O}_{t-s+1} \quad (5)$$

Note that since

$$\sum_{\tau=1}^{D'} P_{t,i,j}(\tau)\sum_{s=1}^{\tau}\boldsymbol{O}_{t-s+1} = \sum_{\tau=1}^{D'}\sum_{s=1}^{\tau} P_{t,i,j}(\tau)\boldsymbol{O}_{t-s+1}$$

$$= \sum_{s=1}^{D'}\sum_{\tau=s}^{D'} P_{t,i,j}(\tau)\boldsymbol{O}_{t-s+1} = \sum_{s=1}^{D'}\boldsymbol{O}_{t-s+1}\sum_{\tau=s}^{D'} P_{t,i,j}(\tau),$$

(5) can be written as follows:

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \sum_{t=1}^{T}\sum_{s=1}^{\min(D,t)} \boldsymbol{O}_{t-s+1}W_{t,i,j}(s) \quad (6)$$

$$\text{where } W_{t,i,j}(s) = \sum_{\tau=s}^{\min(D,t)} P_{t,i,j}(\tau)$$

$$= \sum_{\tau=s}^{\min(D,t)} \alpha_{t-\tau}(i)a_{i,j}d_{i,j}(\tau)\beta_t(j)U_{t,i,j}(\tau)$$

The partial products, $U_{t,i,j}(\tau), \tau = 1,..,\min(D,t)$, can be calculated recursively with the same recursion used in section IV. Later in this section, we will provide a new recursion that reduces the complexity of calculating the scalar weights, $W_{t,i,j}(s), s = 1,..,\min(D,t)$. With the use of these two recursions, we will show that re-estimation is linear in D. Although the method given in section IV is also linear in D, the recursions given in this section have the advantage of using fewer vector operations. We will show that the new method is proportional to M^2+D, compared to M^2D for the previous method.

The summations in 6 can be visualized as an array, where each cell is a contribution to be included in the new estimate of the mean. This array is shown in figure 2. Since each observation vector is multiplied by D different scalar weights, greater efficiency can be achieved by collecting all the scalars together before vector math is used. This can be accomplished by summing along the diagonals of the array shown in figure 2. To see this, we first switch the order of summation:

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \sum_{s=1}^{D}\sum_{t=s}^{T} \boldsymbol{O}_{t-s+1}W_{t,i,j}(s)$$

[3]Note that if $b_{i,j}(O_t)$ and $d_{i,j}(\tau)$ do not depend on the predecessor state i (i.e., a Moore model), then the complexity is $O(NTM^2(K+D-1))$. See the forward and backward definitions in [5].

$s=4$				$\boldsymbol{O}_1 \times W_{4,i,j}(4)$	$\boldsymbol{O}_2 \times W_{5,i,j}(4)$	$\boldsymbol{O}_3 \times W_{6,i,j}(4)$	$\boldsymbol{O}_4 \times W_{7,i,j}(4)$	$\cdots$
			$\boldsymbol{O}_1 \times W_{3,i,j}(3)$	$\boldsymbol{O}_2 \times W_{4,i,j}(3)$	$\boldsymbol{O}_3 \times W_{5,i,j}(3)$	$\boldsymbol{O}_4 \times W_{6,i,j}(3)$	$\cdots$	$\boldsymbol{O}_{T-2} \times W_{T,i,j}(3)$
		$\boldsymbol{O}_1 \times W_{2,i,j}(2)$	$\boldsymbol{O}_2 \times W_{3,i,j}(2)$	$\boldsymbol{O}_3 \times W_{4,i,j}(2)$	$\boldsymbol{O}_4 \times W_{5,i,j}(2)$	$\cdots$	$\boldsymbol{O}_{T-2} \times W_{T-1,i,j}(2)$	$\boldsymbol{O}_{T-1} \times W_{T,i,j}(2)$
$s=1$	$\boldsymbol{O}_1 \times W_{1,i,j}(1)$	$\boldsymbol{O}_2 \times W_{2,i,j}(1)$	$\boldsymbol{O}_3 \times W_{3,i,j}(1)$	$\boldsymbol{O}_4 \times W_{4,i,j}(1)$	$\cdots$	$\boldsymbol{O}_{T-2} \times W_{T-2,i,j}(1)$	$\boldsymbol{O}_{T-1} \times W_{T-1,i,j}(1)$	$\boldsymbol{O}_T \times W_{T,i,j}(1)$
	$t=1$							$t=T$

Fig. 2. Contributions to the mean of $b_{i,j}$. In this example, $D = 4$. Note that the same observation vector is in every cell along a diagonal. The cells to the left of the bold diagonal correspond to the first part of (7); the cells to the right correspond to the second half of (7).

Now let $t' = t - s + 1$:

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \sum_{s=1}^{D} \sum_{t'=1}^{T-s+1} \boldsymbol{O}_{t'} W_{t'+s-1,i,j}(s)$$

Finally, switch the order of summation, rewrite the summation bounds, and bring the observation vector outside the s loop:

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \sum_{t'=1}^{T} \boldsymbol{O}_{t'} \underbrace{\sum_{s=1}^{\min(D,T-t'+1)} W_{t'+s-1,i,j}(s)}_{W\_sum_{t',i,j}}$$

The scalar $W\_sum_{t',i,j}$ can be calculated once and saved before it is multiplied by each of the M components of the observation vector. Because the t^{th} observation vector will contribute to the mean from t until $t + D - 1$, a delay of $D - 1$ is needed, i.e., let $t = t' + D - 1$. This leads to the form of (5) that is most suitable for implementation:

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \overbrace{\sum_{t=D}^{T} \boldsymbol{O}_{t-D+1} \sum_{s=1}^{D} W_{t-D+s,i,j}(s)}^{\text{combined with forward}} + \sum_{t'=T-D+2}^{T} \boldsymbol{O}_{t'} \sum_{s=1}^{T-t'+1} W_{t'+s-1,i,j}(s) \qquad (7)$$

Both time and space efficiency can be improved by combining re-estimation with the forward algorithm.[4] In this case, the first $T-D+1$ observation vectors will be processed at the end of the forward loop. An additional loop is needed to take into account the last $D-1$ observation vectors. Figure 2 shows the division of labor suggested by (7). Assuming that the backward probabilities have already been calculated, consider the following four steps for each iteration of the forward loop:

1. As before, use (1) to recursively calculate $U_{t,i,j}(\tau), \tau = 1, .., \min(D, t)$. These terms are the partial products of output probabilities that form a part of the scalar weights, $W_{t,i,j}(s)$. This step requires $O(M^2 + D)$ operations.

2. Use backward recursion to calculate $W_{t,i,j}(s)$, $s = \min(D, t), .., 1$ (see (6)) in $O(D)$ operations.

$$W_{t,i,j}(s) = \sum_{\tau=s}^{\min(D,t)} P_{t,i,j}(\tau) = P_{t,i,j}(s) + W_{t,i,j}(s+1)$$

$$= \alpha_{t-s}(i) a_{i,j} d_{i,j}(s) \beta_t(j) U_{t,i,j}(s) + W_{t,i,j}(s+1)$$

$$\text{for } \; s = \min(D, t), .., 1$$

$$\text{where } W_{t,i,j}(s) = 0 \text{ for } \; s > \min(D, t)$$

   Even though $U_{t,i,j}(D)$ takes all D observations into account, re-estimation would still be $O(D^2)$ without this step because *each* of the D observations must be weighted by $U_{t,i,j}(D)$. In fact, the first observation term must be weighted by all D values of $U_{t,i,j}(\cdot)$, but their contributions can be collected in $O(D)$ time using the backward sum, $W_{t,i,j}(1)$.

3. If $t \geq D$, sum all of the scalar weights that affect $\boldsymbol{O}_{t-D+1}$. This requires $O(D)$ operations.

$$W\_sum_{t-D+1,i,j} = \sum_{s=1}^{D} W_{t-D+s,i,j}(s)$$

4. If $t \geq D$, update the mean in $O(M)$ operations.

$$\hat{\boldsymbol{\mu}}_{i,j}^{num} = \hat{\boldsymbol{\mu}}_{i,j}^{num} + \boldsymbol{O}_{t-D+1} \times W\_sum_{t-D+1,i,j}$$

$$\hat{\mu}_{i,j}^{den} = \hat{\mu}_{i,j}^{den} + W\_sum_{t-D+1,i,j}$$

By delaying vector operations until all D scalar weights have been accumulated, the complexity of the four steps given above is $O(M^2 + D)$. These steps are repeated for all i, j, and t, leading to the complexity of $O(NKT(M^2 + D))$ for re-estimating the mean. The covariance terms can easily be included in the following additional step:

[4] It is equally advantageous to combine re-estimation with the backward algorithm instead.

5. If $t \geq D$, update the covariance estimate in $O(M^2)$ operations.

$$\hat{\boldsymbol{\Sigma}}_{i,j}^{num} = \hat{\boldsymbol{\Sigma}}_{i,j}^{num} + \boldsymbol{O}_{t-D+1}\,\boldsymbol{O}_{t-D+1}^{*} \times W\_sum_{t-D+1,i,j}$$

$$\hat{\Sigma}_{i,j}^{den} = \hat{\Sigma}_{i,j}^{den} + W\_sum_{t-D+1,i,j}$$

Thus the overall complexity of re-estimation is $O(NKT(M^2+D))$.

The maximum allowed duration need not be the same for each transition. Using a transition-dependent maximum duration, $D_{i,j}$, results in complexity $O(NKT(M^2+D_{avg}))$, where D_{avg} is the average value of $D_{i,j}$ over all i,j. For example, we choose $D_{i,j}$ as the smallest integer such that $\sum_{\tau=1}^{D_{i,j}} d_{i,j}(\tau) \geq 1-\epsilon$, where $\epsilon = 0.01$ in our implementation. To accommodate durations longer than $D_{i,j}$, we combine implicit and explicit duration models in sequence. We essentially add a tail to the duration probability mass function (pmf) by fixing the self transition probability at a small fixed constant. The result is that the overall duration pmf is the convolution of $d_{i,j}$ with a spike-like geometric pmf.

The improved time complexity comes at the cost of increased space complexity. However, if we assume that re-estimation has been combined with the forward algorithm, then the memory needed to store the intermediate variables can be greatly reduced. Since there are only D values of t for which $W\_sum_{t,i,j}$ must be saved, a circular buffer (e.g., using the mod function) of size NKD is needed. This buffer is the same size as the buffer that holds the duration probabilities, $d_{i,j}(\tau)$. Also, it may not be necessary to simultaneously allocate $W\_sum_{t,i,j}$ for all combinations of i and j. It is common in practice to train using a sentence model which is a concatenation of word or phone models. In this case, the number of active output distributions is much smaller than the number of output distributions in the complete model. Also, it is not necessary to save the $W_{t,i,j}(s)$'s if the $W\_sum_{t,i,j}$'s are accumulated in a different way. Consider the following replacement to step 3:

3′. Update the weights for the D most recent observations in $O(D)$ operations.[5]

$$W\_sum_{t-r+1,i,j} = W\_sum_{t-r+1,i,j} + W_{t,i,j}(r)$$

$$\text{for } r = 1,..,\min(D,t)$$

Note that $U_{t,i,j}(\tau), \tau = 1,..,\min(D,t)$ and $W_{t,i,j}(r), r = 1,..,\min(D,t)$ are temporary variables that can be overwritten after the $W\_sum_{t,i,j}$'s have been updated using step 3′. Thus, $U_{t,i,j}(\tau)$ can be replaced by $U(\tau)$ and $W_{t,i,j}(r)$ can be replaced by $W(r)$, requiring negligible space (both buffers are of size D).

VI. Conclusion

We have pointed out that explicit duration modeling has complexity that is linear rather than squared in D, the maximum duration. We have also offered a new recursion that reduces the complexity of training from $O(NKTM^2D)$ to $O(NKT(M^2+D))$.

Tables II and III compare the number of operations for the two methods of re-estimating the covariance matrices. Both tables assume that the forward and backward probabilities have already been calculated.[6] The symmetry of the matrices has been invoked to reduce the number of matrix operations by approximately a factor of two. From the tables, it can be seen that the speedup is approximately $D/2$ if $D \ll M^2$. For typical values of N, K, T, M, and D (see tables II and III), the new method has an order of magnitude fewer multiplications and additions.

References

[1] C.D. Mitchell, M.P. Harper, L.H. Jamieson, and R.A. Helzerman. A parallel implementation of a hidden Markov model with duration modeling for speech recognition. *Digital Signal Processing, A Review Journal*, 5(1), January 1995.

[2] S.E. Levinson. Continuously variable duration hidden Markov models for automatic speech recognition. *Computer Speech & Language*, 1(1):29–45, 1986.

[3] L.R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In A. Waibel and K.F. Lee, editors, *Readings in Speech Recognition*, pages 267–296. Morgan Kaufmann Publishers, Inc., 1990.

[4] P. Ramesh and J. Wilpon. Modeling state durations in hidden Markov models for automatic speech recognition. In *Proc. ICASSP*, pages 381–384, April 1992.

[5] J.D. Ferguson. Variable duration models for speech. In J.D. Ferguson, editor, *Proc. Symposium on the Application of Hidden Markov Models to Text and Speech*, pages 143–179, Princeton, NJ, 1980.

[6] C.D. Mitchell and L.H. Jamieson. Modeling duration in a hidden Markov model with the exponential family. In *Proc. ICASSP*, pages 331–334, April 1993.

[5] We assume that each of the $W\_sum_{t,i,j}$'s was initialized to zero before the forward loop.

[6] For the method corresponding to table III, the additional cost of the forward algorithm is negligible due to the similarity between $W_{t,i,j}(1)$ and $\alpha_t(j)$.

Term	Number of Multiplications	Millions of Multiplications	Number of Additions	Millions of Additions
$\boldsymbol{O}_t\boldsymbol{O}_t^*$	$T(M^2+M)/2$	0.0975	0	0
$b_{i,j}(\boldsymbol{O}_t)$	$NKT(M^2+3M)/2$	525.0	$NKT(M^2+3M)/2$	525.0
$U_{t,i,j}(\tau)$	$NKTD$	37.5	0	0
$\boldsymbol{V}_t^{\Sigma}(\tau)$	0	0	$TD(M^2+M)/2$	2.4375
$P_{t,i,j}(\tau)$	$4NKTD$	150.0	0	0
$\hat{\boldsymbol{\Sigma}}_{i,j}^{num}$	$NKTD(M^2+M)/2$	12,187.5	$NKTD(M^2+M)/2$	12,187.5
$\hat{\Sigma}_{i,j}^{den}$	0	0	$NKTD$	37.5
Total	$O(NKTM^2D)$	12,900.0975	$O(NKTM^2D)$	12,752.4375

TABLE II

COMPLEXITY OF RE-ESTIMATING THE COVARIANCE MATRICES USING THE STANDARD RECURSIONS (SECTION IV). THE NUMERICAL VALUES RESULT FROM THE FOLLOWING ASSUMPTIONS: $N = 2500$, $K = 2$, $T = 300$, $M = 25$, AND $D = 25$.

Term	Number of Multiplications	Millions of Multiplications	Number of Additions	Millions of Additions
$\boldsymbol{O}_t\boldsymbol{O}_t^*$	$T(M^2+M)/2$	0.0975	0	0
$b_{i,j}(\boldsymbol{O}_t)$	$NKT(M^2+3M)/2$	525.0	$NKT(M^2+3M)/2$	525.0
$U_{t,i,j}(\tau)$	$NKTD$	37.5	0	0
$W_{t,i,j}(s)$	$4NKTD$	150.0	$NKTD$	37.5
$W\_sum_{t,i,j}$	0	0	$NKTD$	37.5
$\hat{\boldsymbol{\Sigma}}_{i,j}^{num}$	$NKT(M^2+M)/2$	487.5	$NKT(M^2+M)/2$	487.5
$\hat{\Sigma}_{i,j}^{den}$	0	0	NKT	1.5
Total	$O(NKT(M^2+D))$	1,200.0975	$O(NKT(M^2+D))$	1,089.0

TABLE III

COMPLEXITY OF RE-ESTIMATING THE COVARIANCE MATRICES USING THE NEW METHOD (SECTION V). THE NUMERICAL VALUES RESULT FROM THE FOLLOWING ASSUMPTIONS: $N = 2500$, $K = 2$, $T = 300$, $M = 25$, AND $D = 25$.