A Multi-Resolution Hidden Markov Model using Class-Specific Features

Dr. Paul M. Baggenstoss
Naval Undersea Warfare Center
Newport RI, 02841
401-832-8240 (TEL)
p.m.baggenstoss@ieee.org
EDICS: DSP-APPL, MLR-PATT, MLR-BAYL, MLR-APPL, MLR-GRKN

Abstract

We apply the PDF projection theorem to generalize the hidden Markov model (HMM) to acomodate multiple simultaneous segmentations of the raw data and multiple feature extraction transformations. Different segment sizes and feature transformations are assigned to each state. The algorithm averages over all allowable segmentations by mapping the segmentations to a "proxy" HMM and using the forward procedure.

I. Introduction

A. Motivation

The Hidden Markov Model (HMM) [1] combined with spectral analysis using cepstral coefficients [2] on fixed-length analysis windows (frames) remains at the forefront of automatic speech recognition (ASR) technology. One problem with this architecture is the necessity of using a fixed analysis window size. The need for a fixed-size window arises from the fundamental probabilistic approach that underlies the method and depends on the comparison of likelihood functions formed on a common feature space. In speech and other natural processes, the various phenomena that are being tested (such as phonemes in speech) may occur with differing time scale and require differing frequency resolution to characterize.

This work was supported by the Office of Naval Research (contr nr. N0001405WR20125).

The window size used on speech analysis is a compromise between phonemes with long time scale such as vowels and phonemes with shorter time scales such as plosives. Constant frame-rate processing has two related shortcomings - first the inadequate handling of various-size phonetic units and second, the inadequate modeling of inter-frame dependence (correlation).

B. Goal

Our goal is to develop a statistical framework for (1) combining multiple front-end signal processors using differing window sizes and differing signal processing methods, and (2) solving the segmentation problem simultaneously with the classification problem (not in series) by averaging over all possible sementations.

C. Previous Work

There have been many papers addressing shortcomings of the HMM including techniques such as segmental HMMs or semi-HMMs and graphical models [3], [4], [5], [6], [7], [8], [3], [9], [10]. These techniques offer new statistical topologies that allow for the idea that a Markov state can generate random-length segments. In segmental models, features are typically extracted from groups of frames. These models, however, typically use fixed frame-rate processing as a first step and/or require prior knowledge of the segment boundaries (or some candidate segment boundaries). Some methods get closer to our goal by modeling inter-frame correlation or supplement segment features with multi-resolution features obtained from larger frames or segments formed from multiple frames [11], [12], [13], [14]. While these are steps in the right direction, they fall short of the goal of a completely general statistical framework for combining multi-resolution front-end signal processors.

D. Enabling Theory

With the introduction of the class-specific feature theorem [15], [16], [17], and later the probability density function (PDF) projection theorem (PPT) [18], the freedom now exists to use a different feature set for each class, even for each state in a HMM [19], and as we now show, different analysis window lengths for each state. Thus, the topic of this paper is to apply the PPT to the problem of using varying-size analysis windows within the framework of a HMM.

II. Mathematical Definitions

A. System Resolution and the Elemental Segment

Let T be the basic system resolution. Thus, the finest sub-divisions of data we consider are segments of T samples (elemental segments). In the context of ASR, the elemental segment would be about $T = 1$ or 2 ms to provide adequate time resolution to model short events such as plosives like the "T-burst". Let the input data time-series consist of KT samples

$$\mathbf{X} \triangleq \{x_i,\ \ 1 \leq i \leq KT\}.$$

The input $\mathbf{X}$ represents an extended time-period of time-series data such as a phoneme consisting of several sub-phoneme units, a word, or sentence.

B. Background: HMM with fixed frame rate

Consider a hidden Markov model (HMM) with M Markov states. Let the HMM operate on features extracted from the input data by segmenting the data in uniform segments of T samples. In the context of fixed frame-rate HMM, we can assume that T is about 10 ms. In practice, the segments are usually overlapped but we will assume with no loss of generality that the frames are not overlapped and are not windowed. For the input data of KT samples, there will be exactly K time steps. Let

$$\mathbf{Z} = \{\mathbf{z}_t,\ 1 \leq t \leq K\}$$

be the feature vectors extracted from the K data segments. Let $\{b_q(\mathbf{z}_t),\ 1 \leq q \leq M\}$ be the state observation densities. Let $\mathbf{q} = \{q_1, q_2, \ldots q_K\}$ be a particular state sequence where

$$1 \leq q_t \leq M,\ 1 \leq t \leq K.$$

The assumption usually adopted in HMM applications is that the observations $\mathbf{z}_t$ are conditionally independent (given the states). Therefore, the conditional likelihood function given the state sequence may be written

$$p(\mathbf{Z}|\mathbf{q}) = \prod_{t=1}^{K}\ b_{q_t}(\mathbf{z}_t).$$

The average likelihood function is defined as

$$p_{\text{hmm}}(\mathbf{Z}) = \sum_{\mathbf{q} \in \mathcal{Q}}\ p(\mathbf{Z}|\mathbf{q})\ P(\mathbf{q}),$$

where $\mathcal{Q}$ is the set of all state sequences $\mathbf{q}$. The probability of each state sequence $\mathbf{q}$ is calculated using the state initial probabilities

$$\pi = \{\pi_i,\ 1 \leq i \leq M\},$$

and the state transition probabilities

$$\mathbf{A} = \{A_{i,j},\ 1 \le i \le M,\ 1 \le j \le M\}.$$

In particular,

$$P(\mathbf{q}) = \pi_{q_1} \prod_{t=2}^{K} A_{q_{t-1},q_t}.$$

In HMM applications, we are typically interested in solving the following problems:

1) Efficiently calculating the average likelihood $p_{\text{hmm}}(\mathbf{Z})$. This problem is solved by the forward procedure [1], which has processing load linear in K.
2) Determining the *a posteriori* state probabilities

   $$\gamma_{i,t},\ 1 \le i \le M,\ 1 \le t \le K,$$

   which are defined as the probability that the system is in state i at time t given all the feature data $\mathbf{Z}$. This problem is efficiently solved by the forward procedure combined with the backward procedure [1].
3) Re-estimation of the HMM parameters. Parameters include $\pi, \mathbf{A}$, and the parameters of the densities $b_i(\mathbf{z})$. This problem is efficiently solved by the Baum-Welch algorithm [1].
4) Determination of the optimum state sequence and the associated maximum likelihood

   $$\max_{\mathbf{q}\in\mathcal{Q}}\ p(\mathbf{X}|\mathbf{q})\ P(\mathbf{q}).$$

   This problem is efficiently solved by the Viterbi algorithm [1].

The efficient and convenient properties of the forward procedure, Baum-Welch algorithm and Viterbi algorithm have made the HMM the method of choice in statistical modeling of feature sequences such as found in human speech applications. The disadvantages, as we have pointed out, are due to the need to use the same feature set and segment size for all M states.

C. Definition of the MRHMM.

In a hidden Markov process, the state of the system transitions according to a Markov process and the observed data at a given time is drawn from a distribution that depends only on the identity of the state at that time. In a standard HMM implementation, the sequence of observations are features derived from fixed uniform segmentations of the raw data. The raw data record is synchronized (in time step) with the feature sequence. In other words, the amount of raw data generated for each step of the Markov process is constant. In a multi-resolution HMM (MRHMM), the amount of raw data generated is random. The

process of generating observations is represented by a generative model consisting of two hidden and one observed steps, (1) generation of random discrete Markov state or "sub-class" (hidden), (2) generation of raw data segment size conditioned on the chosen sub-class (hidden), and (3) generation of a raw data segment conditioned on the chosen sub-class and segment size (observed).

Although at a higher level,the MRHMM can be seen as a graphical model [10], or in particular, a segmental HMM [3], the MRHMM is entirely distinct from these methods. Aside from some specialized multi-resolution transforms, e.g. [13], researchers have applied segmental HMMs and graphical models on top of constant frame-rate processing, combining frames to make segments. What distinguishes the MRHMM is (a) the fact that segment boundaries are not observed, and (b) the use of the raw data as common space upon which to compare likelihoods and (c) the use of partial PDF values to cast the problem to a fixed frame-rate HMM.

Note that for the MRHMM, we use the term "sub-class" instead of "state" to avoid possible confusion because we will see later there are three potential definitions of "state". For the purpose of discussion, "sub-class" takes the identity of the smallest phonetic unit. In the context of ASR, it is a segment of data with consistent spectral character such as a number of glottal periods of a vowel or a component of a plosive such as "T-burst". The observed data is the concatenation of the generated raw data segments with the segment boundaries hidden to the observer.

Unlike the conventional HMM implementation where the sequence of frame features are the observations or segmental HMMs where the segment features are observations, the raw time-series data is considered the observation space. While the MRHMM also uses feature transformations, they are sub-class and segment size dependent. The feature space is used only in conjunction with the PDF projection theorem (PPT) as a way to facilitate the calculation of the raw data likelihood functions. For practical reasons, we impose restrictions on the segment sizes. First, we consider only segment sizes in multiples of T samples, the basic system resolution, and second, we assume that segment sizes are selected from a sub-class dependent set of allowable segment sizes.

1) Front-End Processing: The front-end processing of the MRHMM is to realize all possible combinations of segment sizes and feature extraction methods at all time-shifts in multiple of T samples. The log-likelihood values of each segment must also be evaluated (see section II-C7). This processing load is potentially huge. The processing of highly overlapped frames (perhaps 50 ms frames shifted by 1 or 2 ms) can be helped by recursive processing (subtracting the effects of the dropped samples and adding the effect of the added samples) [20]. Note that in this paper we assume all window sizes are shifted at the elemental rate, but in practice, larger window sizes can be substantially down-sampled by inserting

"dummy" frames (frames with negative infinity log-likelihoods).

2) Segmentation: The MRHMM is closely related to the problem of segmenting the input data. However, rather than seeking one "optimum" segmentation or a small number of candidate segment boundaries, the MRHMM considers all segmentations at the same time and can average over all possible segmentations. By a "segmentation" we mean the choice of a contiguous set of segments spanning the K input elemental segments and the assignment of sub-class identity to each segment. We adopt a special notation for a segment of the input data. Let data segment $\mathbf{x}_t^k$ be the length kT segment starting at sample $1+(t-1)T$. Thus, $\mathbf{x}_3^{16}$ is the length $16T$ segment starting at sample $2T+1$ (third elemental segment).

3) Segment-Size Restrictions: For practical reasons, we restrict the segment sizes available for each sub-class. For each signal sub-class i, let there be a set $\mathcal{K}_i$ of integers that represent the segment lengths (in terms of length-T samples) allowed for each sub-class. For example if $\mathcal{K}_2 = \{16, 24, 32, 48, \ldots\}$, then sub-class 2 is assigned possible segment lengths $16T$, $24T$, and $32T$, etc.

Clearly, the dwell time (the total length of time that the system remains in a given sub-class) must be made up from the members of $\mathcal{K}_i$. Since there is no way to predict the exact length of a given input signal event (phonetic unit), the designer must be careful to include a variety of segment sizes so that any potential dwell time can be constructed from a sum of available members. Small segment sizes can act as "fillers" to bridge between larger segments. While it is prudent to include small segment sizes to allow finer resolution in possible dwell times, it is wise to set a minimum segment size to agree with minimum observed dwell time for a given sub-class. Minimum dwell time is set by restricting which segment sizes can be the first segment of the dwell. Define $\mathcal{E}_i$ as the "entry" flags (logical variables equal to 1 or 0 corresponding to whether each segment size in $\mathcal{K}_i$ is allowed to be first). Thus, if $\mathcal{K}_2 = \{4, 6, 10\}$ and $\mathcal{E}_2 = \{0, 1, 1\}$, subclass 2 must begin with a segment of either $6T$ or $10T$ samples and the minimum dwell time for sub-class 2 is 6 time steps ($6T$ samples).

4) Initial and Transition Probabilities: As in the HMM, we define the $M \times M$ sub-class transition probability matrix $\mathbf{A}$ and the $M \times 1$ sub-class initial probability vector π. These two quantities do not, however, explain the generation of segment sizes. For this purpose we define the segment-size policy.

5) Segment size policy: We call the probabilities of random generation of segment sizes a "policy" because it is wise to set these probabilities according to a predefined formula or "policy" rather than attempt to estimate them. While entirely possible and within our framework, estimating the probabilities greatly adds to the number of parameters and is generally not needed. A reasonable policy is a uniform distribution among the available member of $\mathcal{K}_i$ or a policy to favor longer segments. Define the segment size probabilities $\lambda_{i,m}^1$ and $\lambda_{i,m}^0$ for sub-class i. Index m indicates the m-th element of the set $\mathcal{K}_i$. The

superscript 0 indicates transitions within a sub-class, while superscript 1 indicates transitions between sub-classes. We differentiate $\lambda^1_{i,m}$ and $\lambda^0_{i,m}$ because only $\lambda^1_{i,m}$ must be zero if $\mathcal{E}_i(m)$ is zero. Note that in general

$$\sum_m \lambda_m = 1.$$

6) Valid Segmentations: Let S be a particular segmentation of the KT input samples defining the number and order of segments, the sub-class identities of the segments, and their sizes. Let $n(S)$ be the number of segments. Let the sub-class identities be denoted by

$$\mathbf{r}(S) \triangleq \{r_1, r_2, \ldots r_{n(S)}\}.$$

Let

$$\mathbf{m}(S) \triangleq \{m_1, m_2, \ldots m_{n(S)}\}$$

be the indexes of the selected members of of the segment size sets. Let the segment lengths be denoted by

$$\mathbf{k}(S) \triangleq \{k_1, k_2, \ldots k_{n(S)}\}.$$

Thus,

$$k_l = \mathcal{K}_{r_l}(m_l).$$

For S to be a valid segmentation of K elemental segments, we must have

$$\sum_{l=1}^{n(S)} k_l = K. \tag{1}$$

Note that S must also meet the starting segment restrictions set by entry flags $\mathcal{E}_i$.

7) Raw-data segment likelihood functions: Assume for the moment that we are able to compute the data likelihood functions for each sub-class i and each data segment size $k \in \mathcal{K}_i$ and each segment start time t (see section II-C1). Let these likelihood functions be denoted by

$$L_i(\mathbf{x}_t^k) = \log(p(\mathbf{x}_t^k|i)),\ 1 \le i \le M,\ k \in \mathcal{K}_i,\ 1 \le t \le K.$$

Any segment that extends past the KT input samples is not realizable. These segment log-likelihoods can simply be set to minus infinity.

8) Conditional Data Likelihood Function: The likelihood function of the raw data $\mathbf{X}$ given segmentation S assumes conditional independence:

$$p(\mathbf{X}|S) = \exp\left\{\sum_{l=1}^{n(S)} L_{r_l}(\mathbf{x}_{t_l}^{k_l})\right\},$$

where t_l is the starting time (elemental segment number) of the l-th segment.

9) MRHMM average Data Likelihood Function: The average MRHMM likelihood function is defined by

$$p_{\text{mrhmm}}(\mathbf{X}) = \sum_{S \in \mathcal{S}} p(\mathbf{X}|S)\ P(S),$$

where $\mathcal{S}$ is the set of all valid segmentations (see section II-C6), $P(S)$ is the *a priori* probability of segmentation S (to be described later in section II-D6), and

$$\sum_{S \in \mathcal{S}} P(S) = 1.$$

10) Class-Specific Features: Above, we introduced the raw data segment likelihood functions $L_i(\mathbf{x}_t^k) = \log(p(\mathbf{x}_t^k|i))$, but have not explained how they are obtained or how the parameters of these likelihood functions can be estimated. As we alluded to earlier, we apply the PDF projection theorem (PPT)[18]. As we have written several publications on the topic including the tutorial article [21], we describe the method only briefly. Let $\mathbf{x}$ be an arbitrary segment of raw time-series data of length kT samples. Let $\mathbf{z}_i = T_i(\mathbf{x})$ be a feature set calculated from $\mathbf{x}$ specifically designed for sub-class i. Let $b_i^k(\mathbf{z}_i)$ be a PDF estimate of the feature set $\mathbf{z}_i$ based on training data from sub-class i using segments of length k. The feature likelihood function is *projected* from the feature space to the raw data by pre-multiplying by the J-function as follows:

$$p_p(\mathbf{x}|i) = J(\mathbf{x}; T_m, H_{0,i})\ b_i(\mathbf{z}_i). \tag{2}$$

The J-function $J(\mathbf{x}; T_m, H_{0,i})$ is a unique function of $\mathbf{x}$ determined precisely from the feature transformation T_i and the class-dependent reference hypothesis $H_{0,i}$:

$$J(\mathbf{x}; T_i, H_{0,i}) = \frac{p(\mathbf{x}|H_{0,i})}{p(\mathbf{z}_i|H_{0,i})}. \tag{3}$$

Since $J(\mathbf{x}; T_i, H_{0,i})$ is determined *a priori* without regard to training data, it can be considered the *untrained* part of $p_p(\mathbf{x}|i)$ not subject to the curse of dimensionality, while $b_i(\mathbf{z}_i)$ is the trained part.

The function $p_p(\mathbf{x}|i)$ can be regarded as a function only of $\mathbf{x}$ by substituting $T_i(\mathbf{x})$ for $\mathbf{z}_i$ and can be shown to integrate to 1 over $\mathbf{x}$ (thus it is a PDF). While it is true that $p_p(\mathbf{x}|i)$ is a PDF, it is only an estimate of $p(\mathbf{x}|i)$. The degree to which $p_p(\mathbf{x}|i)$ is a good estimate of $p(\mathbf{x}|i)$ depends on many factors including the choice of features and reference hypothesis. Kay [16] has shown the optimality in the K-L distance sense of this choice of PDF.

The J-function takes many forms [18], one of which can be used when $\mathbf{z}_i$ are maximum likelihood (ML) estimates of a set of parameters. In this case, $J(\mathbf{x}; T_i, H_{0,i})$ has a simple form based on the Fisher's information matrix [18].

11) Objectives: In MRHMM applications, we have the following objectives analogous to the objectives in HMM applications.

1) Determine the average likelihood function $p_{\text{mrhmm}}(\mathbf{X})$,
2) Determination of the *aposteriori* sub-class probabilities γ_s^t defined as the probability that sub-class s is true at elemental segment t given $\mathbf{X}$.
3) Re-estimation of model parameters from training data. Parameters include the transition and initial probabilities as well as the parameters of the feature PDFs $b_i^k(\mathbf{z}_i)$.
4) the determination of the best segmentation

$$\max_{S \in \mathcal{S}} \; p(\mathbf{X}|S) \; P(S).$$

Note that this problem is not new and methods exist based on the Viterbi algorithm [22], [23]. But, published applications are primarily based on uniformly segmented data producing feature streams, however the extension to raw data is straight-forward.

D. Implementation of the MRHMM

Now we map the MRHMM to a "proxy" HMM operating at the elemental segment rate that has the same structure as the MRHMM. Each wait state is mapped to a state of the proxy HMM.

1) Mapping of Segmentation to Proxy HMM or Wait State Sequence: Let a *wait state* be defined as an integer that uniquely identifies the state (sub-class), the segment length, and a counter that increments within a segment. To uniquely identify all combinations, there needs to be

$$N_{tot} = \sum_{i=1}^{M} \sum_{k \in \mathcal{K}_i} \; k$$

wait states. Consider the example of $M = 2$, $\mathcal{K}_1 = \{1, 2\}$, $\mathcal{K}_2 = \{2, 4\}$, $\mathcal{E}_1 = \{0, 1\}$, $\mathcal{E}_2 = \{0, 1\}$. The nine wait-states for this example are listed in table I. Each segmentation $\mathcal{S}$ can be uniquely represented by the corresponding length-K sequence of wait-states $\mathbf{w}(S)$.

The proxy HMM is an expanded and restricted HMM that has a parallel structure to the MRHMM. Each wait state of the MRHMM corresponds to a state of the proxy HMM. The proxy HMM requires a special state transition matrix (STM) denoted by $\mathbf{A}^e$ (e stands for expanded) to reflect the special way that wait states behave. A notional example is shown in table II corresponding to the example of table I. An illustrative example for synthetic data is shown in Figure 2. Practical applications may have dozens or even hundreds of wait-states. Notice that the matrix is divided into partitions that correspond Asterisk (*) indicates a potential non-zero element. to members of $\mathcal{K}_i$, that is, available segment length of each

Wait State	Sub-class	Segment Size	Entry	Counter
1	1	1	0	0
2	1	2	1	0
3	1	2	0	1
4	2	2	0	0
5	2	2	0	1
6	2	4	1	0
7	2	4	0	1
8	2	4	0	2
9	2	4	0	3

TABLE I

EXAMPLE WAIT-STATE TABLE FOR A SIMPLE EXAMPLE. PRACTICAL APPLICATIONS MAY HAVE HUNDREDS OF WAIT-STATES.

sub-class. This is why we call the available segment lengths "partitions". Within a partition, wait-states are only allowed to increment to the next wait state until the system reaches the end of the partition. At this point, it may transition to the first wait state of any other partition of a different sub-class that has a non-zero entry flag, or any partition of the current sub-class. Note that due to the entry flags, the STM in the example only allows entry into the length-2 partition of sub-class 1 and the length-4 partition of sub-class 2. Thus, the system will have a minimum dwell time of 2 for sub-class 1 and 4 for sub-class 2. For the purpose of discussion, speech applications may have minimum segment lengths of 20 or 30 ms for vowels and as short as 1 or 2 ms for components of plosives.

2) State Transition Probabilities: The STM of the proxy HMM, $\mathbf{A}^e$, has many elements that are identically zero or one. This is because within a partition, the wait states are forced to advance by one. Otherwise, the system may only transition from the last wait-state of a partition to the first wait-state of a partition. The non-zero elements are determined from the transition probabilities $\mathbf{A}$ and the segment size policy $\lambda_{i,m}$. For transitions within sub-class, e.g. from sub-class i to the m-th partition of sub-class i, the transition probability is $A_{i,i}\ \lambda^0_{i,m}$. For transitions between sub-classes, e.g. from sub-class i to the m-th partition belonging to sub-class j, the transition probability is $A_{i,j}\ \lambda^1_{j,m}$ (this assumes entry flag $\mathcal{E}_j(m)$ is true). An example of a proxy STM is shown in table II and agrees with the example in table I.

3) Initial Probabilities: The initial probability matrix for the proxy HMM, π^e, is partitioned similar to a row of the STM and is straight-forward to construct from the initial probabilities π, and the segment size policy $\lambda^1_{i,m}$. In general, system may only start in the first wait-state of a partition and only if the

0	*	0	0	0	*	0	0	0
0	0	1	0	0	0	0	0	0
0	*	0	0	0	*	0	0	0
0	0	0	0	1	0	0	0	0
0	*	0	0	0	*	0	0	0
0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	1
0	*	0	0	0	*	0	0	0

TABLE II

STATE TRANSITION MATRIX (STM) CORRESPONDING TO THE EXAMPLE OF TABLE I. ASTERSK (*) INDICATES A POTENTIAL NON-ZERO ELEMENT.

entry flag is true. For initialization at the m-th partition belonging to sub-class i, the initial probability is $\pi_i\ \lambda_{jim}^1$ (this assumes entry flag $\mathcal{E}_i(m)$ is true).

4) Partial PDF values: The proxy HMM has N_{tot} states and accordingly a STM of size $N_{tot} \times N_{tot}$. To operate the forward procedure of the proxy HMM, we will need a (possibly very large) likelihood matrix of $N_{tot} \times K$ state likelihood values analogous to $b_i(\mathbf{z}_t)$ in the conventional HMM. As the MRHMM operates in the raw data space, these likelihood values are obtained from the raw data segment likelihoods $L_i(\mathbf{x}_t^k)$. But there is a dilemma here because the number of required likelihood values far exceeds what is available from the raw data segment likelihoods. To fill in this likelihood matrix, each raw data segment log likelihood $L_i(\mathbf{x}_t^k)$ is divided by k and replicated k times to obtain what we call the "partial PDF values". They are *partial* PDF values because alone they are meaningless. They are meaningful only when the k partial values are added up, recovering $L_i(\mathbf{x}_t^k)$. To be more specific, let $P_{w,t}$ be the partial PDF value for wait state $w,\ 1 \leq w \leq N_{tot}$ and time $t,\ 1 \leq t \leq K$. Let

$$P_{w,t} = L_i(\mathbf{x}_u^k)/k.$$

Notice that t is the elemental segment number of the proxy HMM and u is the starting elemental segment the segment in question. Note that

$$u = t - c$$

where c is the wait state counter value (see table I). Notice that the partial probability matrix is redundant and need not be stored explicitly. An illustrative example for synthetic data is shown in Figure 3 and for

a real-data example in Figure 7.

5) Efficient Programming: Both the number of states and the number of time-steps of the proxy HMM are much larger than the corresponding sizes in a typical HMM implementation (the elemental segment size of the MRHMM is typically much smaller than the frame rate of an HMM). This can cause excessive computational load. But, as we will see later, the sparsity of the proxy STM provides ample opportunity for efficient programming.

6) Forward Procedure on the Proxy HMM: We state our main theorem.

Theorem 1: Let the proxy HMM be constructed from the MRHMM consisting of the initial probabilities as in section II-D3, the state transition probabilities as in section II-D2, and the partial probabilities in section II-D4. Then the forward procedure on the proxy HMM calculates the MRHMM average probability $p_{\text{mrhmm}}(\mathbf{X})$.

The correctness of the theorem can be seen as follows. Let

$$\mathbf{w}(S) = [w_1, w_2, \ldots w_K]$$

be any state trajectory on the proxy HMM corresponding to a valid segmentation S. Summing the partial PDF values along trajectory $\mathbf{w}(S)$ will serve to accumulate all the partial PDF values of each segment in the segmentation S, forming $p(\mathbf{X}|S)$. The trajectory probability $P(\mathbf{w})$ serves as the segment probability $P(S)$. While any trajectory is technically valid, only those trajectories corresponding to a valid segmentation S will have non-zero probability.

III. Training

A. Training state PDFs in the standard HMM.

Training a HMM is accomplished by the Baum Welch re-estimation algorithm [1], which is a special case of an expectation-maximization (EM) algorithm. The part of the Baum Welch algorithm that re-estimates the state PDFs is simply a weighted maximum likelihood update. That is, it attempts to maximize the weighted log-likelihood function of each state. The weight changes at each time update and is equal to $\gamma_{i,t}$, the *a prosteriori* probability (given all the data) that state i is in effect at time t. The values of $\gamma_{i,t}$ are obtained by running the *forward* and *backward* algorithms and multiplying the results [1]. Let

$$L_i(\mathbf{Z}) = \sum_{t=1}^{N} \gamma_{i,t} \ \log p(\mathbf{z}_t|i)$$

be the weighted log-likelihood for state i. Maximizing this function is often straight-forward. For example when $p(\mathbf{z}|i)$ takes the form of a multivariate Gaussian distribution, updating the mean and covariance is as simple as using the $\gamma_{i,t}$-weighted mean and covariance estimates.

B. Training state PDFs in the MRHMM.

We cannot apply the Baum Welch algorithm to the proxy HMM because it is expanded from a smaller set of MRHMM parameters. With no further modification of the Baum Welch algorithm, there would need to be a PDF estimate for each wait state. However, for the MRHMM, there are only PDF estimates associated with each partition Logically, the MRHMM produces values of $\gamma_{w,t}$ that are constant along time diagonals in a partition. That is,

$$\gamma_{w,t} = \gamma_{w+1,t+1}$$

if wait states w and $w+1$ are in the same partition. Thus, in the MRHMM, each analysis window can be traced to a given constant-valued diagonal line in the $\gamma_{w,t}$ matrix. To train the MRHMM, the features from the associated window are weighted by the corresponding value of $\gamma_{w,t}$ in the diagonal.

C. Ganging States.

While each partition is associated with a PDF estimate, we may not want to train all partition PDF estimates independently. For example, if a high value of $\gamma_{w,t}$ indicates that a large partition is active, it is advantageous to train the shorter partitions of the same sub-class on the same data. To do this, we "gang together" all partitions that associate with a given sub-class. To gang partitions, we first create a compressed version of $\gamma_{w,t}$, denoted by $\gamma^c_{i,t}$, which sums $\gamma_{w,t}$ over all wait states associated with sub-class i. Then we then weight an analysis window by the *smallest* value of $\gamma^c_{i,t}$ in the set of time steps t contained in the analysis window. This works very well in practice but is a clear departure from the Baum-Welch algorithm and may produce an algorithm without guaranteed monotonicity.

D. Training Initial and Transition Probabilities.

In the fixed frame-rate HMM, the Baum-Welch algorithm re-estimates the initial and transition probabilities at each update [1]. It essentially works by from $\gamma_{i,t}$ by adding up the appropriate opportunities weighted by the probability of each opportunity. Applied to $\gamma_{w,t}$ of the the proxy HMM directly, the algorithm provides estimates of "total weighted opportunity" for each element of the $N_{tot} \times N_{tot}$ matrix $\mathbf{A}^e$. It is straight-forward to compress these results to obtain estimates of $\mathbf{A}$ by adding up the partitions of each sub-class. Similarly, estimates of π^e are obtained from the first column of $\gamma_{w,t}$ and need to be compressed to obtain estimates of π.

E. Model Order Selection

The MRHMM is not immune to the model order selection problem. That is, an increasing number of parameters in a model invariably leads to better apparent fit to the data. Thus, segmentations that have large numbers of small segments may dominate. There are two ways to mitigate this problem. The natural way is to use more parameters for longer segments. This is natural because longer segments have finer frequency resolution. Thus, use higher autoregressive model order or larger numbers of cepstral coefficients for the longer segments. A second way is to apply a penalty function that compensates for higher model order such as minimum description length (MDL) In general, the MDL criterion [24] is implemented by subtracting the term $\frac{P}{2}\log N$ from the log-likelihood value. Consider a segmentation S defined for $N = KT$ input samples. Let $\{L_{r_l}(\mathbf{x}_{t_l}^{k_l}),\ 1 \le l \le n(S)\}$ be the segment log-likelihood values for segmentation S (see section II-C8) and let $p_1, p_2 \ldots p_{n(S)}$ be the feature dimensions for the feature sets extracted from each segment. The MDL criterion is implemented by subtracting $\frac{p_l}{2}\log(KT)$ from each segment log-likelihood (It is incorrect to use $\frac{p_l}{2}\log(k_l T)$). This compensates $p(\mathbf{X}|S)$ for the total number of parameters used in segmentation S

$$P(S) = \sum_{l}^{n(S)} p_l.$$

But this leads to the following dilemma: as the input data record size KT increases, so does the relative amount of compensation applied to each segment. For this reason, we have found it best to use a fixed value of N set to approximately the largest segment size.

IV. Simple Illustrative Example

To illustrate the concepts, we tested the concept of the MRHMM using an over-simplified example of simulated data. To independent identically distributed (iid) Gaussian noise, we added a low frequency (LF) pulse of autoregressive (AR) process of 128 samples in length with a peak frequency response of 0.4 radians per sample, followed by high frequency (HF) pulse of AR process of 64 samples with a peak frequency response of 1.2 radians per sample. There was a random-length gap of noise between them. An example of the signal and noise is shown in Figure 1.

We implemented the MRHMM with three sub-classes: “noise”, “LF pulse”, and “HF pulse”. We used a total of nine partitions expanding to a total of 25 wait states. The elemental segment length was $T = 32$ samples. Parameters of the nine partitions are listed in table III. The expanded state transition matrix is shown in figure 2.

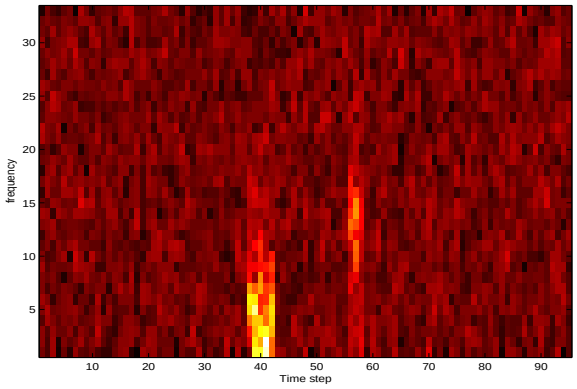

Fig. 1. Example of spectrogram of synthetic data. The data consists of three sub-classes. Class 1 (noise) occurs first, then a low-frequency pulse of duration 128 samples, then noise, then a high-frequency pulse of duration 64 samples.

Partition	Signal class	Length (samples)	K	AR order P
1	Noise	256	8	4
2	Noise	128	4	4
3	Noise	64	2	4
4	Noise	32	1	3
5	LF Pulse	128	4	4
6	LF Pulse	64	2	4
7	LF Pulse	32	1	3
8	HF Pulse	64	2	4
9	HF Pulse	32	1	3

TABLE III

PARTITION PARAMETERS FOR THE ILLUSTRATIVE EXAMPLE.

Autoregressive features of model order P (see table I) were extracted by overlapped window processing. A separate feature processor was used for each combination of K and P. Analysis windows were shifted always by the elemental segment length of 32 samples for each update, so the amount of overlap depended on the length of the analysis window. To handle end effects, data was assumed to wrap around in time.

Features were extracted from each analysis window by first taking the FFT, computing the magnitude squared, then computing the inverse-FFT to produce the autocorrelation function (ACF). The Levinson algorithm was used to produce the reflection coefficients of order P. The total power in window is also stored as the $P+1$st feature. The J-function [18] is obtained by use of the saddle-point approximation

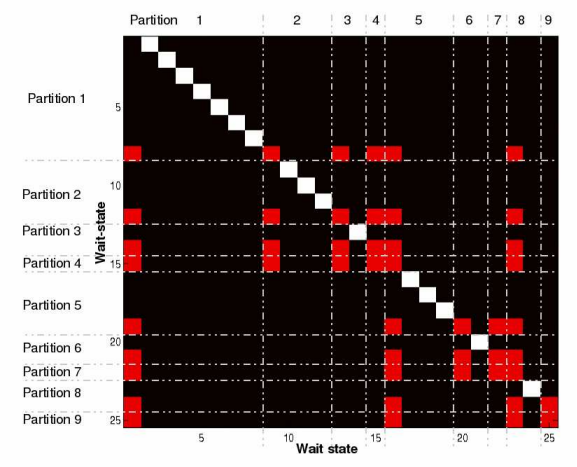

Fig. 2. State transition matrix $\mathbf{A}^e$ (expanded to include wait states)

[25]. Further details of the implementation details of the AR models can be found in [21].

In Figure 3, we see the partial PDF matrix $P_{i,m}$ for a typical sample. Wait states 1 through 15 are associated with the "Noise" sub-class. wait states 16 through 22 are associated with the "LF Pulse" sub-class, and wait states 23 through 25 are associated with the "HF Pulse" sub-class.

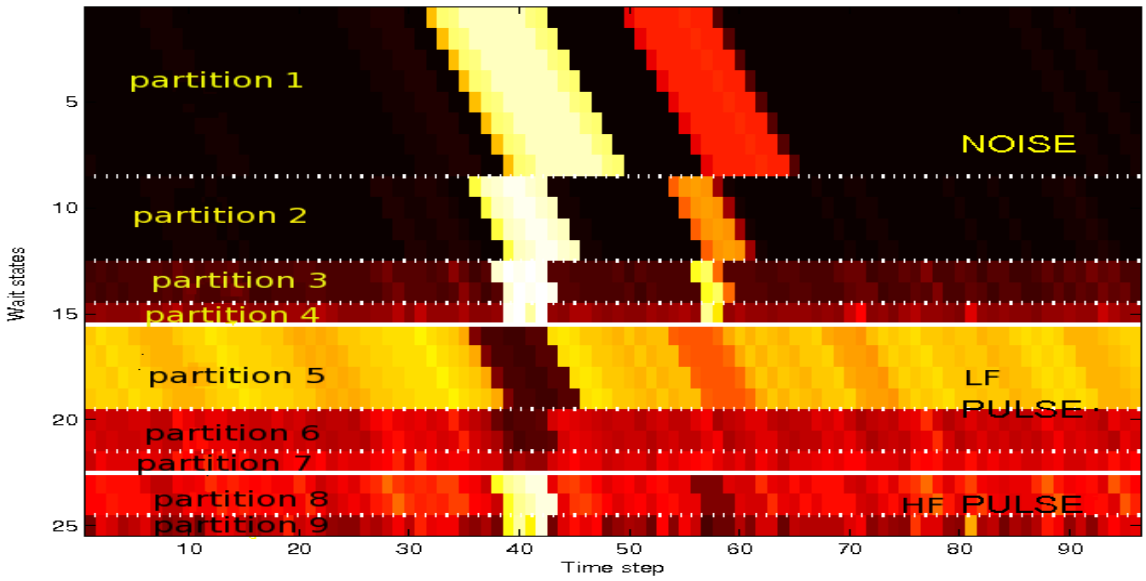

Fig. 3. Partial PDF matrix $P_{w,t}$ showing devisions between sub-classes (solid horizontal lines) and between wait state partitions (dotted lines). Higher probability is darker.

The gamma probabilities are a by-product of the Baum Welch algorithm [1] and indicate the relative probability of each wait state given the data. The gamma probabilities corresponding to figure 3 are shown in Figure 4. This figure can be interpreted as the paths through figure 4 that pick up the highest probabilities while meeting the restrictions set by the state transition matrix. Note that the wait states

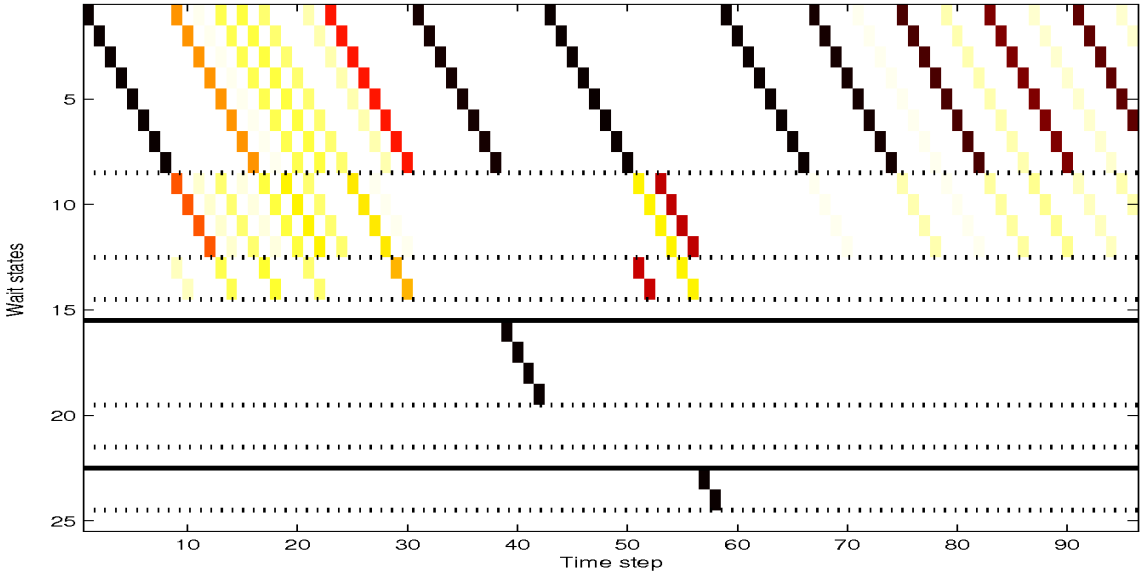

Fig. 4. Wait state probabilities for illustrative example.

for “LF pulse” (16 through 19) are clearly seen where the pulse occurs. The same is true of the “HF pulse” event (23 through 24) . It is possible to see various competing paths through the trellis. Note for example in time steps 43 through 56, the noise gap between the two pulses, the HMM is in the noise sub-class. In steps 43-50, it is in partition 1, (wait states 1 through 8). Then after exiting wait state 4, it has located two possibilities to span the six time steps remaining before HF pulse occurs. It can either go into partition 2 (wait states 9 through 12) then partition 3 (wait states 13 through 14), or it can choose the reverse, partition 3 then partition 2.

The gamma probabilities can be collapsed to indicate just the sub-classes, as shown in Figure 5. The class probabilities (figure 5) is an accurate indication of the true content of the data to a time resolution of $T = 32$ samples.

V. Results using Speech data

We applied the MRHMM algorithm to create a statistical model the word “tan”. In total, we identified eight sub-classes classes in the word “tan”. Including “silence” and “noise”, we allocated sub-classes

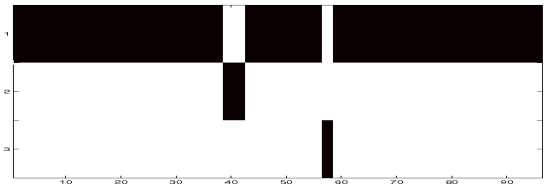

Fig. 5. Signal class probabilities calculated by summing figure 4 over the wait states of each class.

for “T-burst”, “T-aspiration”, “A”, and “N”. Since the word begins with a “T-closure”, there is no stop preceding the closure so “T-closure” is mapped to “silence”. We also allowed two sub-classes for the transition region from “T” to “A”. The sample rate was 12 kHz and the system resolution was $T = 16$ samples. Window sizes of $384, 256, 192, 128, 96, 64, 48, 32, 16,$ samples were used ($k = 24, 16, 12, 8, 6, 4, 3, 2, 1$). Autoregressive parameters were extracted with corresponding a model orders of ($P = 8, 7, 6, 6, 5, 4, 3, 3, 2$). The sub-classes are tabulated in table IV along with the segment length sets $\mathcal{K}_i$ and entry flag sets $\mathcal{E}_i$. There are a total of 43 partitions containing a total of 386 wait states. We assumed

Signal Class	$\mathcal{K}_i$	$\mathcal{E}_i$
Silence	[24 16 12 8 6 4 3 2 1]	[1 1 1 0 0 0 0 0 0]
Background Noise	[24 16 12 8 6 4 3 2 1]	[1 1 1 1 1 1 1 1 1]
T-2 (Burst)	[4 3 2 1]	[1 1 1 1]
T-3(Aspiration)	[12 8 6 4 3 2 1]	[1 1 1 1 1 1 1]
T-4.1 (transition 1)	[16 12 8 6]	[1 1 1 1]
T-4.2 (transition 2)	[16 12 8 6]	[1 1 1 1]
A	[24 16 12]	[1 1 1]
N	[24 16 12]	[1 1 1]

TABLE IV

THE EIGHT SUB-CLASSES IN THE WORD “STOOL” AND THEIR SEGMENT SIZE SETS AND ENTRY FLAGS.

a left-to-right structure for the transition matrix (no back-tracking). We trained on 36 recordings of the word “tan” and initialized the algorithm with hand-labeled data, then let it free-run. Results are shown in Figures 6 and 7. In figure 6, the time-series and spectrogram are compared with the probabilistic sub-class mapping ($\gamma_{i,t}^c$). The sub-class probabilities in this sample are extremely crisp showing approximately 0 and 1 everywhere. In figure 7, partial probability matrix $P_{w,t}$ and expanded (wait-state) *a posteriori* state

probabilities $\gamma_{w,t}$ are shown for the same sample. All the components of the "T" can be clearly seen by observing $\gamma^c_{i,t}$.

VI. Conclusions and Future work

We have presented a novel algorithm for statistical signal modeling, analysis and classification. It performs simultaneously the functions of signal processing, segmentation, and classification. It combines the outputs of highly redundant and overlapped signal processing analysis windows to statistically model time-series data in a solid statistical framework. The method offers the ability to select signal processing methods and window size individually for each sub-class. It solves the segmentation problem in a probabilistic approach that simultaneously combines and compares the results of all possibile segmentations. It provides an average likelihood value that sums over all segmentations weighted by the probability of each segmentation. The algorithm can re-estimate model parameters using an extension of the Baum-Welch algorithm.

We have demonstrated the potential of the algorithm as a method of modeling signals and speech waveforms. In modeling speech signals, we have demonstrated the ability of the MRHMM to locate sub-word units with high resolution. It may also allow "surgical precision" in modeling sub-word units. The method has clear potential in ASR but is not mature enough to compare with existing ASR methods. Much work needs to be done to address the usual issues in ASR having to do with the great variability in input waveforms that share the same phonetic information. We have demonstrated the method using autoregressive features. It may be necessary to extend these results to MEL Cepstrum features.

References

[1] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–286, February 1989.

[2] J. W. Picone, "Signal modeling techniques in speech recognition," *Proceedings of the IEEE*, vol. 81, no. 9, pp. 1215–1247, 1993.

[3] M. Ostendorf, V. Digalakis, and O. A. Kimball, "From hmms to segment models: A unifed view of stochastic modeling for speech recognition," *IEEE Transactions on Speech and Audio Processing*, 1996.

[4] A. Kannan and M. Ostendorf, "A comparison of trajectory and mixture modeling in segment-based word recognition," *ICASSP 1993*, Apr 1993.

[5] J. R. Glass, "A probabilistic framework for segment-based speech recognition," *Computer Speech and Language*, vol. 17, pp. 137–152, Oct 2003.

[6] J. N. Marcus and V. W. Zue, "A variable duration acoustic segment hmm for hard-to-recognize words and phrases," *proceedings of ICASSP 1991*, pp. 281–284, Apr 1991.

[7] M. J. F. Gales and S. J. Young, "The theory of segmental hidden markov models," *Technical Report, University of Cambridge*, Jun 1993.

[8] K. Achan, S. Roweis, and B. Frey, "A segmental hmm for speech waveforms," *University of Toronto Technical Report*, Jan 2004.

[9] K. P. Murphy, "Hidden semi-markov models (hsmms)," *Informal Notes*, Nov 2002.

[10] J. A. Bilmes, "Graphical models and automatic speech recognition," *Mathematical Foundations of Speech and Language Processing*, 2003.

[11] S. Vaseghi, N. Harte, and B. Milner, "Multi-resolution phonetic/segmental features and models for hmm-based speech recognition," *ICASSP 1997*, vol. 2, 1997.

[12] V. Digalakis, M. Ostendorf, and J. R. Rohlicek, "Improvements in the stochastic segment model for phoneme recognition," *Proceedings of the workshop on speech and natural language*, pp. 332–338, 1989.

[13] S. Vaseghi, N. Harte, and B. Milner, "Multi-resolution phonetic/segmental features and models for hmm-based speech recognition," *ICASSP 1997*, vol. 2, 1997.

[14] Y.-S. Yun and Y.-H. Oh, "A segmental-feature hmm for continuous speech recognition based on a parametric trajectory model," *Speech Communication*, vol. 38, pp. 115–130, September 2002.

[15] P. M. Baggenstoss, "Class-specific features in classification.," in *IASTED International Conference on Signal and Image Processing*, 1998.

[16] S. Kay, "Sufficiency, classification, and the class-specific feature theorem," *IEEE Trans. Information Theory*, vol. 46, pp. 1654–1658, July 2000.

[17] P. M. Baggenstoss, "Class-specific features in classification.," *IEEE Trans Signal Processing*, pp. 3428–3432, December 1999.

[18] P. M. Baggenstoss, "The PDF projection theorem and the class-specific method," *IEEE Trans Signal Processing*, pp. 672–685, March 2003.

[19] P. M. Baggenstoss, "A modified Baum-Welch algorithm for hidden Markov models with multiple observation spaces.," *IEEE Trans. Speech and Audio*, pp. 411–416, May 2001.

[20] P. Baggenstoss, "Time-series segmentation," *United States Patent 6907367*, June 2005.

[21] P. M. Baggenstoss, "The class-specific classifier: Avoiding the curse of dimensionality (tutorial)," *IEEE Aerospace and Electronic Systems Magazine, special Tutorial addendum*, vol. 19, pp. 37–52, January 2004.

[22] T. Svendsen and F. Soong, "On the automatic segmentation of speech signals," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 12, pp. 77–80, 1987.

[23] R. Andre-Obrecht, "A new statistical approach for the automatic segmentation of continuous speech signals," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 36, Jan 1988.

[24] S. Kay, *Fundamentals of Statisticsl Signal Processing, Estimation Theory*. Prentice Hall, Upper Saddle River, New Jersey, USA, 1993.

[25] S. M. Kay, A. H. Nuttall, and P. M. Baggenstoss, "Multidimensional probability density function approximation for detection, classification and model order selection," *IEEE Trans. Signal Processing*, pp. 2240–2252, Oct 2001.

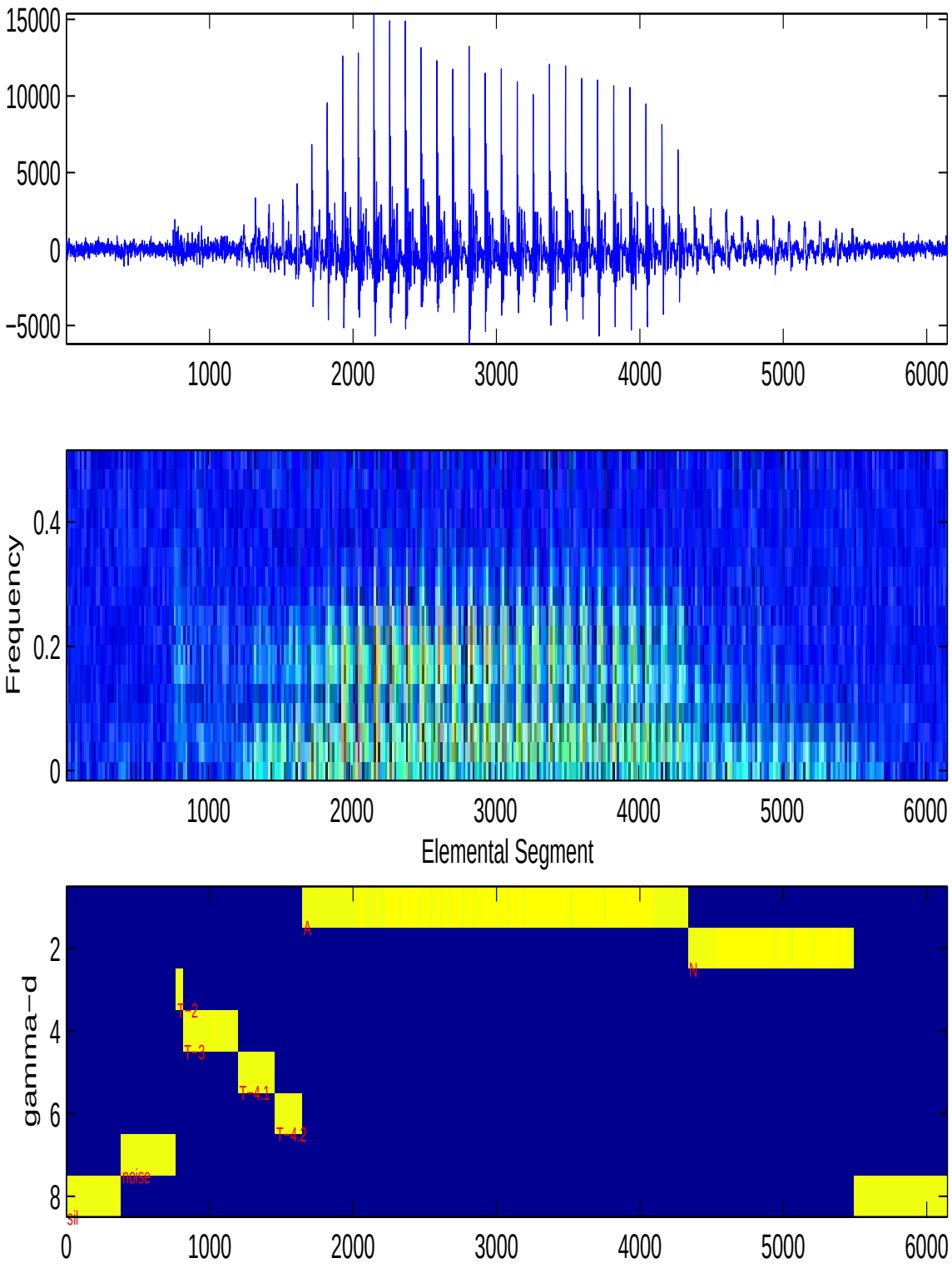

Fig. 6. Example of MRHMM operating on the word “tan”. Top: time-series. Middle: spectrogram. Bottom: compressed *a posteriori* state probabilities $\gamma_{i,t}^{c}$. The burst and aspiration of “T” can be clearly identified.

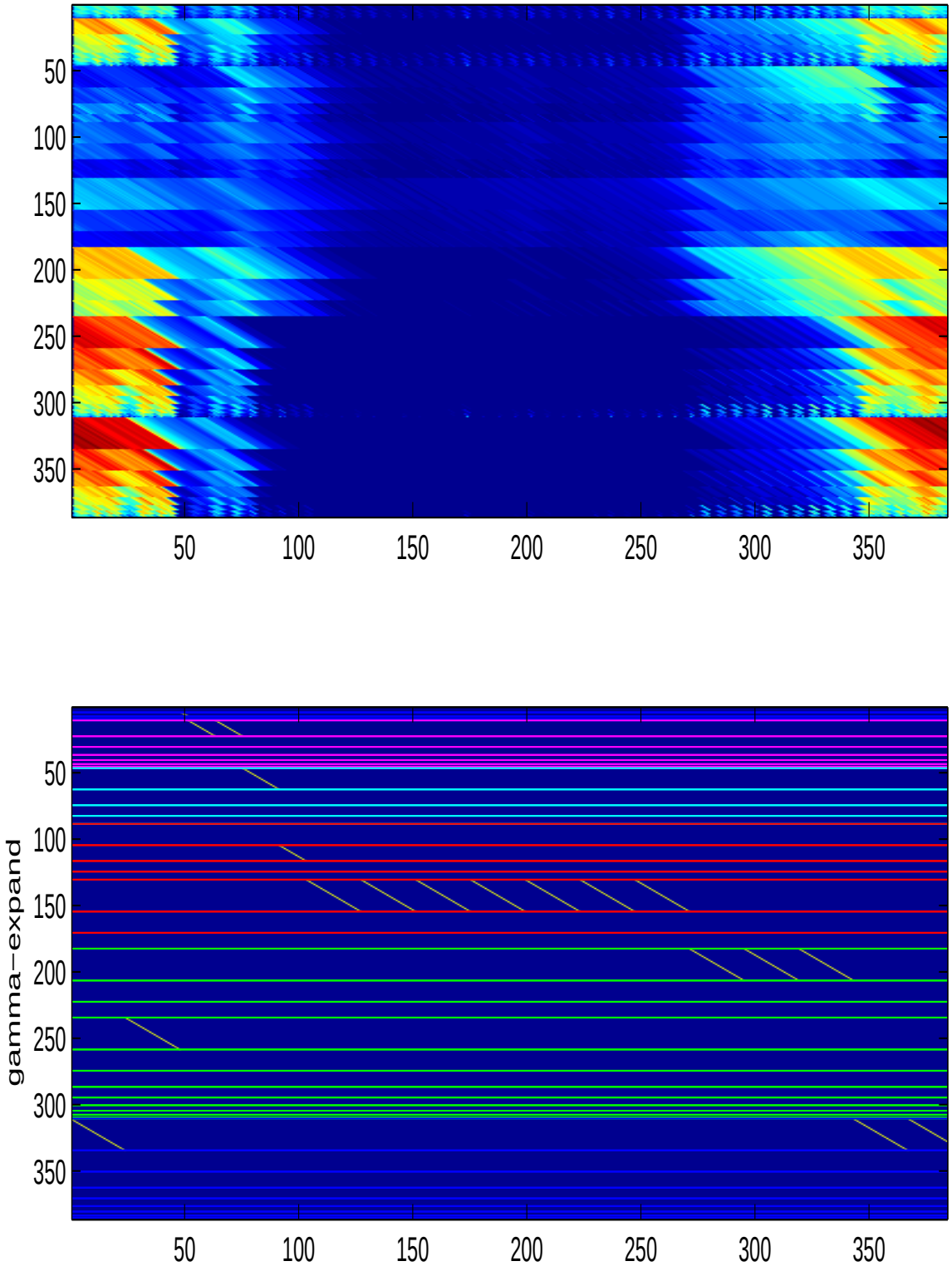

Fig. 7. Example of MRHMM operating on the word “tan”. Top: partial probability matrix $P_{w,t}$. Bottom: expanded (wait-state) *a posteriori* state probabilities $\gamma_{w,t}$.