The Markov Selection Model for concurrent speech recognition

Paris Smaragdis[a,1], Bhiksha Raj[b,1]

[a] *University of Illinois, Urbana Champaign, IL, USA*
Adobe Systems Inc.
[b] *Carnegie Mellon University, Pittsburgh, PA, USA*

1. Introduction

Speech recognition of concurrent speakers is a significantly hard task. Although contrived as a scenario, its solution can be of great use to noise-robust speech recognition and can provide insight on how to deal with some of the hardest problems in acoustic sensing. Current models for speech recognition cannot be easily extended to deal with additive interference, and often need to be complemented with a separation algorithm that preprocesses the data before recognition takes place. This is often a risky combination since the output of a separation algorithm is not guaranteed to be recognizable speech, at least not by a speech recognition system.

A different temporally-sensitive approach characterizes the speech from all concurrent sources (speakers) by HMMs. The sum of the speech is then characterized by a factorial HMM, which is essentially a product of the HMMs representing the individual sources. Inference can be run on this factorial HMM to determine what was spoken by individual speakers. The problem of course is computational complexity: the number of states in the factorial HMM is the product of the states in the HMMs for individual sources, i.e. is polynomial in the number of concurrent sources. The time taken for inference is polynomial in twice the number of sources, requiring complicated variational methods to make it tractable.

In this paper we introduce a *Markov selection model* coupled with a probabilistic decomposition that allows us to recognize additive speech mixtures in time that is *linear* in the number of concurrent sources. We make use of speaker-dependent models, which when used on speech mixtures can obtain reliable estimates of all the spoken utterances. In the following sections we describe an additive model which has been used in the past for source separation, then describe how it can be incorporated into an HMM-based speech recognition framework to form the proposed Markov selection model. Finally we present results from experiments based on the ICSLP 2006 speech separation challenge data (Cooke, 2006).

2. Additive Models of Sounds

Recently we have seen wide use of non-negative factorization methods with applications to source separation from single channel recordings (Raj and Smaragdis, 2005; Virtanen and Cemgil, 2009; Smaragdis, Shashanka, and Raj, 2009). In this paper we will make use of such a model and adapt it for use in a speech recognition system. Let us begin by reviewing how these models work and also setup the notation used henceforth.

Many modern source separation methods use prior knowledge of the sources in a mixture. A common scenario is one where for two speakers, speaker a and speaker b, we have training recordings $x^a(t)$ and $x^b(t)$, and a mixture $m(t) = y^a(t) + y^b(t)$. Our goal is then to use the information extracted from $x^a(t)$ and $x^b(t)$ to estimate $y^a(t)$ and $y^b(t)$ by observing only $m(t)$.

A very efficient and capable approach to perform this task is by using non-negative spectrum factorization methods. Here we will use a probabilistic version of these techniques which allows us to later incorporate it in a Markov model. In this setting, we extract the spectral magnitude of the observed signals at regularly sampled analysis frames:

$$X_\tau(f) \propto ||\,\mathrm{DFT}(x(T(\tau-1)+1,...,T\tau))||, \tag{1}$$

where T is the size of the analysis frame we choose to use. Doing so we can obtain X^a_τ and X^b_τ, the magnitude spectra for signals from speakers a and b. We model magnitude spectra as histograms drawn from a mixture of multinomial distributions. This leads to the following latent variable model:

$$X_\tau(f) \sim \sum_z^M P(f|z)P_\tau(z), \tag{2}$$

where the symbol $\sim$ represents drawing from a distribution, $P(f|z)$ represents the z^{th} component multinomial and $P_\tau(z)$ is the probability with which it is mixed to produce X_τ, the magnitude spectrum vector for the τ^{th} analysis frame. M is the total number of component multinomials. The component multinomials (which we will refer to as "multinomial bases") $P(f|z)$ for any speaker and the corresponding mixture weights $P_\tau(z)$ for each spectral vector can now be estimated using an Expectation-Maximization algorithm.

This is essentially a simplified pLSI model (Hoffmann, 1999), but looking past its probabilistic formulation we note that $P(f|z)$ is in fact a normalized spectrum. The set of all multinomials can thus be viewed as a dictionary of spectral bases, Equation 2 can be viewed as an algebraic decomposition and M as the rank of this decomposition. $P_\tau(z)$ can be seen as weights that tell us how to put the dictionary elements together to approximate the input at hand. Thus Equation 2 can be written as:

$$X_\tau(f) \approx \hat{X}_\tau(f) = g_\tau \sum_z^M P(f|z)P_\tau(z), \tag{3}$$

where $g_\tau = \sum_f X_\tau(f)$. The scalar g_τ ensures that the eventual approximation is scaled appropriately to match the input. This can also be thought of as a non-negative matrix factorization (Lee and Seung, 1999) in which $P(f|z)$ and $P_\tau(z)$ correspond to the two non-negative factors.

There are two key observations we need to make in order to be able to extract $y^a(t)$ and $y^b(t)$ from $m(t)$. The first one is that in general it will hold that:

$$M_\tau(f) \approx Y^a_\tau(f) + Y^b_\tau(f). \tag{4}$$

This means that the magnitude spectrogram of the mixture of the two sources will be approximately equal to the sum of the magnitude spectrograms of the two sources. Although due to phase cancellations we cannot achieve exact equality, this assumption has been used with great success so far and it is largely correct for most practical purposes.

The second observation is that the multinomial bases $P^a(f|z)$, that we can estimate from X^a_τ, can describe Y^a_τ better than the bases $P^b(f|z)$ estimated from X^b_τ, and vice versa, i.e.:

$$\begin{aligned} \mathrm{D_{KL}}\left(\frac{Y^a_\tau}{g_\tau} || \sum_z^M P^a(f|z)P_\tau(z)\right) < \\ \mathrm{D_{KL}}\left(\frac{Y^a_\tau}{g_\tau} || \sum_z^M P^b(f|z)P_\tau(z)\right) \end{aligned} \tag{5}$$

and vice-versa. Where $\mathrm{D_{KL}}(\cdot)$ denotes the Kullback-Leibler divergence, $P^a(f|z)$ and $P^b(f|z)$ are the dictionaries learned from x^a and x^b, and each $P_\tau(z)$ is the optimal weight distribution for approximating Y^a_τ given each of the two dictionaries.

These two observations then allow us to assume that the observed mixture $M_\tau(f)$ can be explained well using both dictionaries $P^a(f|z)$ and $P^b(f|z)$:

$$\begin{aligned} M_\tau(f) \approx g_\tau P_\tau(a) \sum_z^M P^a(f|z)P_\tau(z) \\ + g_\tau P_\tau(b) \sum_z^M P^b(f|z)P_\tau(z), \end{aligned} \tag{6}$$

for two optimally selected instances of $P_\tau(z)$. In addition to being well explained, we would expect to see most of the energy of each speaker being represented by the part of this summation that includes the multinomial bases for that speaker.

For both the dictionary learning and the weight estimation parts, we can use the Expectation-Maximization algorithm to estimate any of the quantities in the above equations. The update equations for any dictionary element $P(f|z)$

and its corresponding weight $P_\tau(z)$ for an input $X_\tau(f)$ are:

$$P_\tau(z) = \frac{\sum_f P_\tau(z|f)X_\tau(f)}{\sum_{z',f} P_\tau(z'|f)X_\tau(f)} \tag{7}$$

$$P(f|z) = \frac{\sum_\tau P_\tau(z|f)X_\tau(f)}{\sum_{z',\tau} P_\tau(z'|f)X_\tau(f)} \tag{8}$$

where:

$$P_\tau(z|f) = \frac{P_\tau(z)P(f|z)}{\sum_{z'} P_\tau(z')P(f|z')}. \tag{9}$$

The dictionary of multinomial bases for each of the sources may be learned from separate training data. These can be used to decompose mixed recordings (i.e. to find the mixture weights $P\tau(z)$ for all bases). Once the decomposition in Equation 6 is achieved, we can then recompose separately $Y_\tau^a(f)$ and $Y_\tau^b(f)$ and then invert them back to the time domain to obtain our separated estimates of $y^a(t)$ and $y^b(t)$. The performance of this approach can vary depending on various details we do not present here, but in its better forms this has been shown to achieve suppression of unwanted speakers up to 20dB or more (Smaragdis et al., 2009).

However since the objective of this paper is to recognize as opposed to separate, it would be more beneficial to use direct recognition using this model as opposed to separating and then recognizing. To do so we will incorporate this model into a hidden Markov model structure as shown in the following section.

3. The Markov Selection Model

3.1. Model definition

In this section we will introduce an application of the model and the observations made in the previous section as applied on temporal data. A conventional Hidden Markov Model is a doubly-stochastic model comprising an underlying Markov chain and observation probability densities at each state in the chain. The parameters characterizing the model are the *initial state probabilities* $\Pi = \{P(s)\forall s\}$ representing the probabilities of beginning a Markov chain at each state, a *transition matrix* $\mathbf{T} = \{P(s_i|s_j)\forall s_i, s_j\}$ that represents the set of all transition probabilities between every pair of states, and a set of *state output distributions* $\mathbf{B} = \{P(x|s)\forall s\}$ representing the probability of generating observations from each of the states. The graphical model for the HMM is shown in Figure 1.**a**.

The model we propose extends this conventional model as shown in Figure 1.**b**. Instead of states generating observations directly, they generate the labels $\mathbf{z}_s = \{z\}$ of sets of multinomial bases that will produce observations. Thus, the output distributions of the HMM are $\mathbf{B} = \{P(\mathbf{z}_s|s)\forall s\}$. To generate observations, the multinomial bases in $\mathbf{z}_s$ are "mixed" according to weights w_z. The vector of weights for all bases, $\mathbf{w}$, which actually represents a multinomial over z, is drawn from a distribution which, for this paper, we will assume is

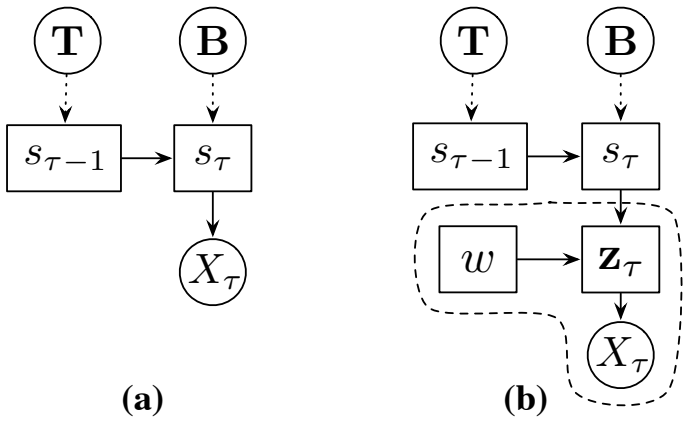

Figure 1: *(***a***) A graphical representation of a conventional HMM. The state at each time is dependent on the state at the previous time and generates the observation. Dotted arrows indicate injection of parameters. (***b***) The proposed Markov selection model. The state selects the multinomial bases that generate the observation. The bases are "mixed" by a weight w. This additional dependence is highlighted by the dotted outline.*

uniform. Only the bases selected by the state (and their weights, appropriately normalized) are used to generate the final observation. Since the underlying Markov process contributes to data generation primarily by *selecting* bases, we refer to this as the *Markov Selection Model*. Figure 2 illustrates the generation process with an example.

A key aspect of the above model is that the weights w_z are not fixed but are themselves drawn for every observation. Further, the draw of the weights themselves is not dependent on the state in any manner, but is independent. The actual probability of an observation depends on the mixture weights. Thus, in order to compute the complete likelihood of an observation we must integrate the product of the weight-dependent likelihood of the observation and the probability of drawing the mixture weight vector over the entire probability simplex on which $\mathbf{w}$ resides.

We note that the primary use for this model is that of inferring the underlying state sequence given this model. To do so, it is sufficient to determine the Markov-chain-independent *a posteriori* probabilities $P_{ind}(s|X)$ of the states for each observation X, and utilize those for estimating the state sequence; the actual observation probability $P(X|s)$ is not required. Indeed, this observation is also utilized in several approaches to HMM-based speech recognition systems where the Markov-chain-independent *a posteriori* probabilities of states are obtained through models such as Neural Networks (Bourlard and Morgan, 1998) for inference of the underlying word sequence.

As a first step, instead of explicitly integrating over the space of all weights to obtain the likelihood of the observation, we will use the Markov-chain-independent *a posteriori* state probability for all inference and learning of Markov chain parameters. Secondly, we will approximate the *a posteriori* state proba-

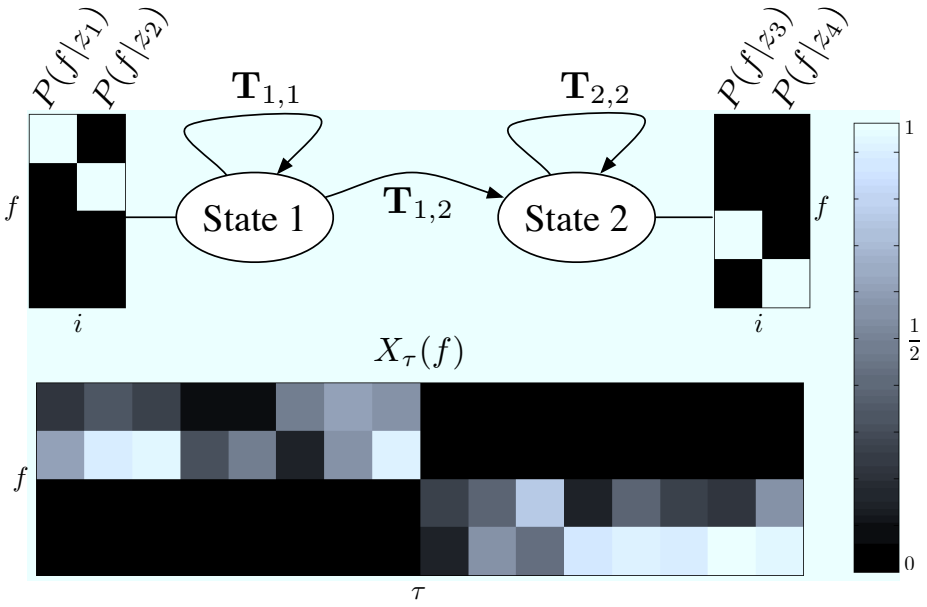

Figure 2: *A two state left-to-right Markov model is shown at the top. Each state selects one pair of multinomial bases. The two bases that describe each state are shown left and right as* $P(f|z_i)$*. The bottom of the figure displays the input* $X_\tau(f)$ *that this model can best describe. The left part being best described as a mixture of* $P(f|z_1)$ *and* $P(f|z_2)$ *and the right part by* $P(f|z_3)$ *and* $P(f|z_4)$*.*

bility by the sum of *a posteriori* most likely mixture weights for the Multinomial bases selected by any state. I.e. we use the approximation:

$$P(z|X_\tau) \approx \hat{P}(z|X_\tau) = \arg\max_{z'} P(z'|x) = P_\tau(z) \tag{10}$$

$$P_{ind}(s|X) = \sum_{z \in \mathbf{z}_s} \hat{P}(z|X_\tau) = \sum_{z \in \mathbf{z}_s} P_\tau(z) = P_\tau(\mathbf{z}_s) \tag{11}$$

where $P_\tau(z)$ is the same value referred to in Equation 8. In other words, we derive the mixture weights that maximize the likelihood of the portion of the graph enclosed by the dashed outline in Figure 1.**b**. We do so without reference to the Markov chain and utilize them to compute the markov-chain-independent conditional probabilities for states, which will be used in the inference. This effectively factors the observation dependency and the state dependency of the model.

Thus, one of the advantages conferred by the approximation is that the model of Figure 1.**b** gets factored in two parts. The first (enclosed by the dashed outline in the figure) is essentially a pLSA model that obtains $\mathbf{w}_{ml}$ and thereby $P_\tau(z)$. The second, given the $P_\tau(z)$ computed from the first part, is effectively an HMM with $P_\tau(\mathbf{z}_z)$ as state output densities. Inference and learning can run largely independently in the two components, with the pLSA component

employed to learn its parameters, while the HMM can use the standard Baum-Welch training procedure (Rabiner and Juang, 1986) to learn the Markov chain parameters Π and $\mathbf{T}$. The two components must however combine for learning the multinomial bases $P(f|z)$.

3.2. Parameter estimation

In order to train the distribution, we observe that the structure of Figure 1 comprises two conditionally independent structures: an upper doubly-stochastic Markov process and a lower latent variable model, which are linked through the variable $\mathbf{z}_\tau$. Given the conditional dependencies shown by the model, and the assumption that $P(s|X) \approx P_\tau(\mathbf{z}_s)$, learning for the upper Markov chain is strictly analogous to learning the parameters of an HMM with state emission probability $P(X|s) = P_\tau(\mathbf{z}_s)$. The parameters of this structure can hence be learned using the Baum-Welch training procedure (Rabiner and Juang, 1986).

In the first step we compute the "emission" probability terms for each state. Since this is locally also a maximum likelihood estimate we estimate an intermediate value of the optimal weight vector by:

$$P_\tau(z|f) = \frac{P_\tau(z)P(f|z)}{\sum_{z'} P_\tau(z')P(f|z')} \tag{12}$$

$$P_\tau(z) = \frac{\sum_f P_\tau(z|f)X_\tau(f)}{\sum_{f,z'} P_\tau(z'|f)X_\tau(f)} \tag{13}$$

Note that the above estimation does not refer to the underlying Markov chain or its states; all computations are local to the components within the dotted outline of Figure 1.**b**. Once $P_\tau(z)$ has been obtained, we compute the posterior state probability $P(s|X_\tau) = P_\tau(\mathbf{z}_s)$ using Equation 11.

The forward backward algorithm can then be employed as in conventional HMM. Forward probabilities α, backward probabilities β and state posteriors γ are given by the recursions:

$$\begin{aligned}
\alpha_\tau(s) &= \sum_{s'} \alpha_{\tau-1}(s')T_{s,s'}P_\tau(\mathbf{z}_s) \\
\beta_\tau(s) &= \sum_{s'} \beta_{\tau+1}(s')T_{s,s'}P_{\tau+1}(\mathbf{z}_{s'}) \\
\gamma_\tau(s) &= \frac{\alpha_\tau(s)\beta_\tau(s)}{\sum_{s'} \alpha_\tau(s')\beta_\tau(s')}.
\end{aligned} \tag{14}$$

In the Maximization step we estimate both the transition probabilities of the Markov chain and the dictionary elements.

The transition probabilities and initial state probabilities are estimated identically to the conventional Baum Welch procedure. The updated transition

probabilities are given by

$$\gamma_\tau(s,s') = \frac{\alpha_\tau(s)T_{s,s'}\beta_{\tau+1}(s')P_{\tau+1}(\mathbf{z}_{s'})}{\sum_{s_1,s_2}\alpha_\tau(s_1)T_{s_1,s_2}\beta_{\tau+1}(s_2)P_{\tau+1}(\mathbf{z}_{s_2})} \quad (15)$$

$$T_{s,s'} = \frac{\sum_\tau \gamma_\tau(s,s')}{\sum_\tau \gamma_\tau(s)} \quad (16)$$

The updated initial state probabilitie $P(s)$ of any state is simply the normalized sum of $\gamma_1(s)$ over all utterances in a training set.

To update the dictionary elements $P(f|z,i)$ we use the state posteriors to appropriately weigh Equation 8 and obtain:

$$P(f|z) = \frac{\sum_\tau \sum_{s:z\in\mathbf{z}_s}\gamma_\tau(s)P_\tau(z|f)X_\tau(f)}{\sum_{f'}\sum_{\tau,s:z\in\mathbf{z}_s}\gamma_\tau(s)P_\tau(z|f')X_\tau(f')} \quad (17)$$

Here "$s : z \in \mathbf{z}_s$" represents the set of states which can select basis z. Update rules for the initial state probabilities are the same as with traditional HMM models.

3.3. Avoiding Local Optima

A common problem when using the above parameter estimation process is that of local optima. As described in the previous section we essentially perform two dependent estimation steps: that of the dictionary elements $P(f|z)$ and that of the temporal statistics $\mathbf{T}$. These estimations are being refined in parallel and proper convergence of both is essential. If we start with random values for the initial estimates of $P(F|z)$ then it is highly likely that one of the dictionary elements that, say, belongs to the last state in the chain might best explain data in the beginning of the input. If this happens, due to the nature of EM training, these dictionary elements will increasingly adapt in every iteration to explain the beginning as opposed to the end of the input. This will in turn bias the estimation of the states and the temporal statistics and force the entire model to quickly go through all the states in the first input frames and converge to the last state for the majority of the input. One of the primary causes of this problem is that adaptation of the dictionary elements is rather rapid, and once near a local optimum it will tend to stay there and not backtrack to correct for a mismatches in the state order.

Convergence to local optima is of course a well known problem when performing optimization with EM and there are no good and efficient principled approaches to address such issues. We can however alleviate this problem by correcting some of the reasons why things can go wrong. As mentioned before one of the problems in convergence arises due to the dictionary elements settling on locally optimal values before the proper state transitions have a chance to form. In order to address this issue we will slow down the adaptation of the dictionary elements so that the implied state transitions settle on a plausible path before the dictionary elements commit to describe specific sections of the input. This can be easily achieved by imposing an "anti-sparsity" prior on the

mixture weights $P_\tau(z)$ of the dictionary elements. We do so by using a Dirichlet prior over $P_\tau(z)$ with hyper-parameters α_z, all gradually transitioning from 1.5 to 1 during training. This simply implies that in every training iteration after we estimate the new value of $P_\tau(z)$ we additionally perform the operation:

$$P_{tau}(z)^* = P_{tau}(z) + (\alpha_z - 1) \tag{18}$$

and then subsequently normalize $P_{tau}(z)^*$ so that it sums to 1 and use it as the final estimate of $P_\tau(z)$ for the current iteration. Adding any positive constant to the current estimate of $P_\tau(z)$ will keep any of its values from dropping to zero and thus all dictionary elements will remain eligible to explain all parts of the input sequence. By slowly decreasing the α_z values so that we ultimately apply no modification to $P_\tau(z)$ we give a chance to the state path to form before the dictionary elements latch on an implausible sequence.

It should be noted however that this is not a foolproof approach and there often some experimentation needed to find the optimal values and decay rate for α_z. Currently this is still one of the weaknesses of this approach, and a good candidate for revision in future research.

3.4. State Sequence Estimation

The procedure for computing the optimal state sequence, given all model parameters, is straight forward. For each observation we compute the emission probability for each state through the EM estimation of Equations 13 and Equation 11. The Viterbi algorithm can then be used to find the optimal state sequence.

The two examples in Figure 4 illustrate the results obtained, both from the learning and the state sequence estimation. For each example, a three-state model of the proposed architecture is learned. We then obtain the optimal state sequence for each data sequence using the model estimated from it. The state segmentations obtained are shown in the bottom plots of Figure 4. As we can observe, the segmentation obtained is intuitive – each of the states captures a locally consistent region of the data.

4. Modeling mixtures of sounds

The main advantage of the proposed model becomes apparent when we use it to analyze the sum of the output of two separate processes. Let $X^a_\tau(f)$ and $X^b_\tau(f)$ be two data sequences obtained separately from two sources that are well modeled by the model of Section 3. Let the actual observation $X_\tau(f) = X^a_\tau(f) + X^b_\tau(f)$. Then it is relatively straightforward to show that the statistical model for $X_\tau(f)$ is given by Figure 3. As before, each of the two sources follows its own independent Markov chain. The state output distributions for each source are selector functions, as in the case of the single source. The primary difference lies in the manner in which the summed data are generated. An independent process now draws a mixture weight vector that *includes mixture weights for all bases of both sources*. The final observation is obtained by the

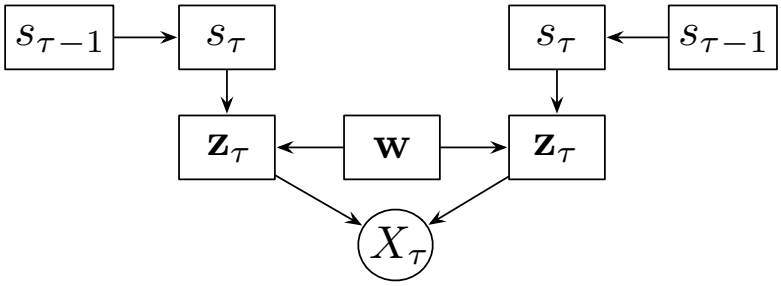

Figure 3: *Model for producing a mixture of two sources. Each source follows its own Markov chain. Each source's state selects some subset of multinomial bases. The mixture weights for all bases (including both sources) are independently drawn. The final observation is generated by the weighted mixture of bases selected by both sources.*

mixing of the bases selected by the states of both of the sources using the drawn mixture weights.

The problem of estimating the state sequences for the individual sources is now easily solved. Using the same approximation we use in Section 3.1 we first compute the optimal weights for all bases using iterations of Equation 13. The iterations compute the $P_\tau(z)$ for *all bases from all sources*. Once these are computed, the Markov-chain-independent *a posteriori* state probabilities for each of the states of the Markov models for both sources are computed using Equation 11 as follows:

$$P(s|X_\tau^{(i)}) = \sum_{z \in \mathbf{z}_s} P_\tau(z) \tag{19}$$

where $X_\tau^{(i)}$ is the i^{th} source at time step τ, s is any state in the Markov model for the i^{th} source and $\mathbf{z}_s$ is the set of bases selected by the state.

Remarkably, the model permits us compute the state emission probabilities for the individual sources, given only the *sum* of their outputs. The optimal state sequences for the individual source can now be independently obtained by the Viterbi algorithm. As a result, the complexity of this process, given K sources, each modeled by N states is $O(KN^2)$, equivalent to performing K independent Viterbi decodes. This is in contrast to conventional *factorial* approach to modeling the mixture of multiple sources, where the resulting model has N^K states and Viterbi estimation of the optimal state sequence requires $O(N^{2K})$ operations necessitating complex variational approximations to simplify the problem.

Figure 5 illustrates the proposed procedure. The top plot is a "mixed" data sequence composed as a sum of the two sequences in Figure 4. Ordinarily in this situation we would have to use a factorial Markov model which would consider all twelve possible combinations between both models' states, and then obtain the most likely state paths using a 2-d Viterbi search. Using the proposed approach we obtain the individual emission scores for the states of the individual HMMs for every time instant and obtain the optimal state sequence independently for

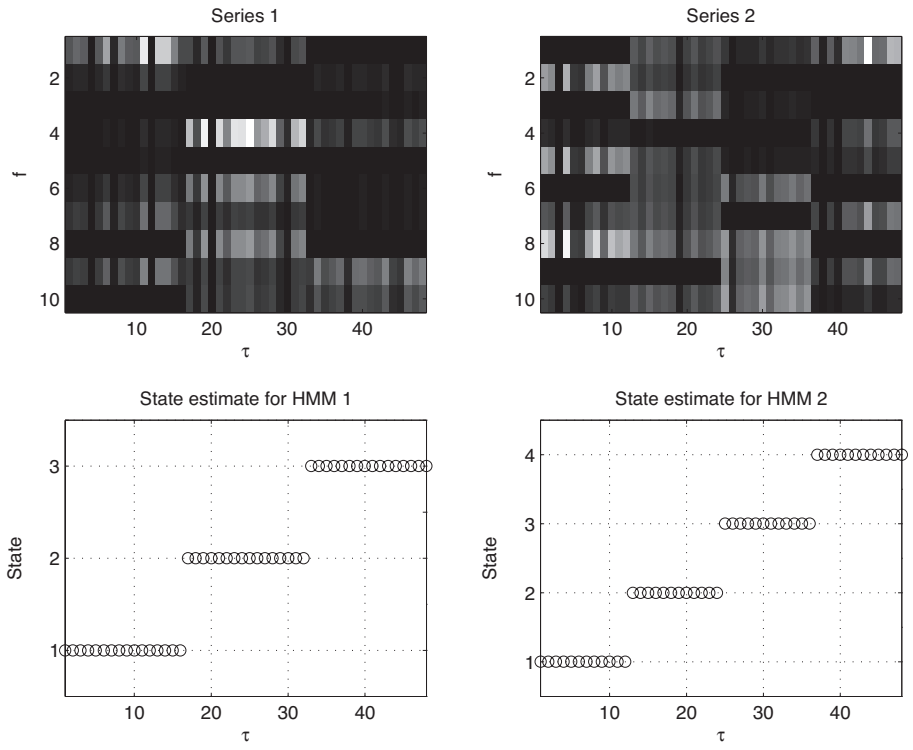

Figure 4: *Two input patterns (top figures) and their corresponding state sequences (bottom figures) as discovered by the proposed model.*

both sources. The obtained state sequences are shown in the bottom plots of Figure 5. We note that they are identical to the state sequences obtained from the isolated sequences in Figure 4. Multiple runs over various inputs provide similar results with only occasional and minor differences between the states extracted from the isolated sequences and their sum.

5. Recognizing Mixtures of Speakers

In this section we present two experiments that demonstrate the use of this model in speech recognition applications. We present a small experiment that performs digit recognition on simultaneous digit speaking by the same speaker, and a larger experiment based on the speech separation challenge data (Cooke, 2006).

5.1. A small scale experiment

In this experiment we use digit data from the speech separation challenge corpus to illustrate the ability of this model to discover sequences from speech

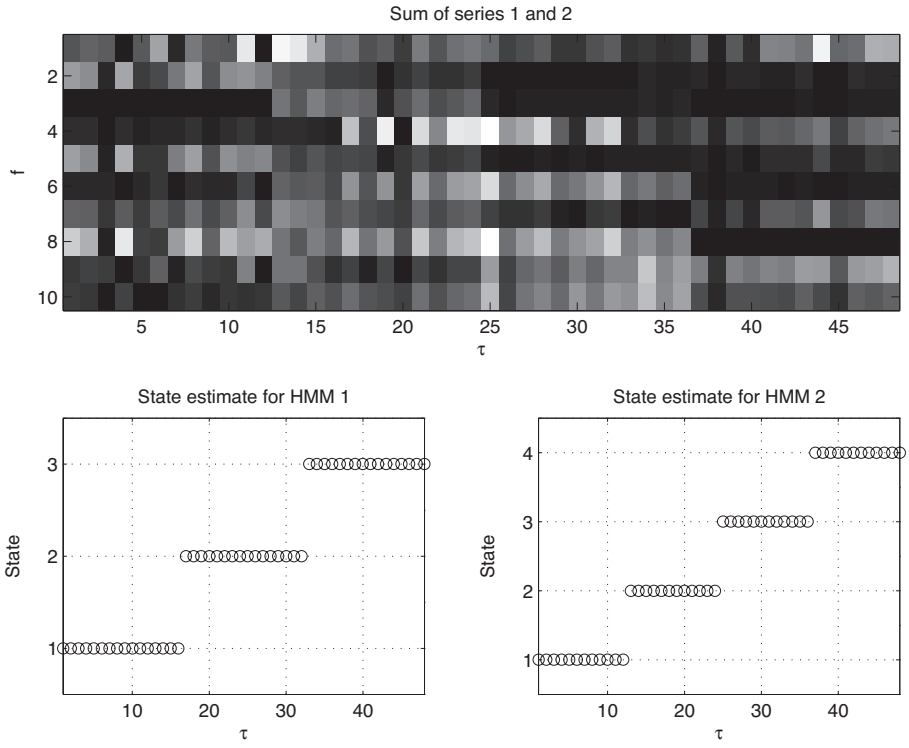

Figure 5: *Sum of the patterns in Figure 4 and the extracted state sequences as computed by the models learned on the isolated patterns.*

mixtures. In this experiment we chose ten utterances of five different digits from one speaker and we trained an instance of the proposed Markov model for each digit. For simplicity we used four states for all digits and three frequency distributions for each state. As an input we used pre-emphasized magnitude spectra from roughly 45ms windows. We then used an additional unknown utterance of each digit to construct a set of sound mixtures containing one digit each. We analyzed these mixtures using the pre-learned digit models and examined their estimated likelihoods in order to discover which utterances were spoken in the mixture. A representative example of these results is shown for four mixture cases in Figure 6. The log likelihoods of the spoken digits were significantly higher than the non-spoken digits and from them we can easily deduce the contents of the recording.

In the next section we generalize this idea to a much larger and more thorough experiment.

5.2. A large scale experiment

In this section we describe an experiment using the speaker separation challenge data set (Cooke, 2006). We present results on both the task set forth for

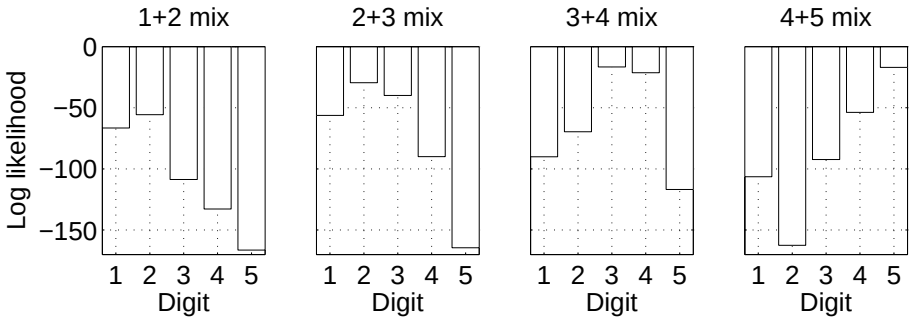

Figure 6: *Estimated model log likelihoods evaluated on mixtures of digits. Each subplot contains the log likelihoods of each digit model for a different digit mixture as denoted above each plot. As shown here the two highest log likelihoods coincide with the digits that were spoken in that mixture, thereby providing us with the digit recognition.*

this challenge, but also for full word recognition. The data in this challenge were composed of mixture recordings of two speakers simultaneously uttering sentences of a predefined structure. In the first case we evaluate the task is to identify a specific word in the sentence uttered by the primary speaker, whereas in the second case we attempt to recognize all words for both utterances.

The features we used were magnitude spectral features. We used a time frame of about 30ms, and a frame advance of 15ms. The magnitude spectra were preemphasized so that the higher frequency content was more pronounced. We trained the proposed model for each word and each speaker using the number of states guidelines provided by the dataset documentation. We used one frequency distribution per state and trained each model for 500 iterations. The resulting models from each speaker were then combined to form a larger Markov model which can model an entire target sentence with equiprobable jumps between all candidate words at each section. For each mixture sentence the speaker identities were provided in advance and the two Markov models describing all the possible utterances were used to estimate the most likely state sequence for each speaker as described in the previous section. The results of these simulations are shown in tables 1, for the proposed task in the challenge, and 2 for the word recognition rates for both of the simultaneous utterances. The SNR column describes the amplitude difference between the primary and the secondary speakers. As expected the louder the primary speaker is the better results we achieve. The "Same speaker" column shows the results when the two utterances were recorded from the same speaker. This is the worst case scenario since the dictionary elements in our model will have maximal overlap and the state posterior probabilities will be unreliable. We clearly see that in this case the results are the lowest we obtain. The "Same gender" column describes the results when the two speakers were of the same gender. This is a somewhat better situation since the will be less overlap between the state dictionary elements and the results reflect that by being higher. Finally the even better case is when the two speakers are of different gender, in which case

there is a high likelihood that dictionary elements will not overlap significantly, thus we obtain the best results. The final two columns present the average results of our proposed model and for a baseline comparison, the "GMM Avg." label presents the average results we obtained using the same representation and a Gaussian state HMM, while treating the secondary speaker as noise.

The overall results we obtain rank high in terms of previously achieved results for this task (Cooke, 2006), and come at a significantly lower computational cost than other approaches due to the efficient decoding scheme we introduce.

SNR	Same speaker	Same gender	Diff gender	Avg.	GHMM Avg.
6dB	58.1%	68.3%	69.8%	65.2%	48.0%
3dB	46.4%	64.2%	64.7%	58.0%	37.2%
0dB	32.7%	53.9%	60.5%	48.6%	29.4%
-3dB	21.7%	44.8%	53.0%	39.3%	20.8%
-6dB	13.6%	36.0%	45.7%	31.2%	15.5%
-9dB	8.7%	31.5%	37.0%	25.2%	12.3%

Table 1: *Detailed task results using the proposed model*

SNR	Same speaker	Same gender	Diff gender	Avg.
Clean	N/A	N/A	N/A	88%
6dB	68%|32%	80%|59%	83%|70%	77%|53%
3dB	57%|42%	77%|67%	80%|76%	71%|61%
0dB	46%|53%	68%|75%	76%|80%	63%|69%
-3dB	35%|65%	61%|80%	71%|84%	55%|76%
-6dB	26%|74%	53%|84%	64%|86%	47%|81%
-9dB	21%|80%	48%|87%	57%|87%	41%|84%

Table 2: *Overall word recognition results using the proposed model. Left percentages are denoting the correct recognition rate for the primary speaker's words, right percentages do so for the secondary speaker.*

An interesting point to make here is that the representation that we used is balancing a tradeoff between mixture modeling and recognition. The fine frequency resolution and linear amplitude scale that we use aid in discriminating the two speakers and facilitates the additivity assumption, but it also impedes recognition since it highlights pitch and amplitude variances. In contrast to that, a speech recognition system would use a lower frequency resolution that conceals pitch information but maintains spectral shape, and would also use that representation in the log amplitude domain so that subtle amplitude patterns can be easier to detect. Selecting the proper representation is a process that involves trading off the ability to discriminate sources and the ability to recognize, something which is application dependent.

Once the state transitions have been estimated from a mixture, it is also

trivial to perform separation of the constituent sources. Since this operation is out of the scope of this paper we defer the presentation of this experiment to future publications.

6. Conclusions

In this paper we introduced the *Markov Selection Model*, a new statistical model which combines models used for source separation with a Markov structure. We demonstrate how this model can learn and recognize sequences, but also perform recognition and state estimation even when presented mixed mixed signals. Superficially this model is similar to the one in (Ozerov, Févotte, and Charbit, 2009), but at closer inspection provides a more extensive basis model and is substantially different in estimation and inference. We formulate this model in such a way that so that it allows us to perform state estimation on mixed sequences with linear complexity in the number of sources. This is a significant computational improvement as compared to similarly employed factorial Markov models, and one that doesn't sacrifice performance by a noticeable amount. This structure can also be represented as a Conditional Random Field, which can result in additional structural possibilities and a more straightforward inference formulation, and it can also be used for solving source separation and denoising problems. We anticipate to address these possibilities in future work.

Acknowledgements

The authors acknowledge Gautham Mysore's help in the preparation of this manuscript.

Bibliography

References

Bourlard, H., Morgan, N., 1998. Hybrid HMM/ANN systems for speech recognition: Overview and new research directions. Lecture Notes in Computer Science 1387, 389–417.

Cooke, M., 2006. Speech separation challenge. `http://www.dcs.shef.ac.uk/~martin/SpeechSeparationChallenge.htm`.

Hoffmann, T., 1999. Probabilistic latent semantic indexing. In: ACM SIGIR Special Interest Group on Information Retrieval Conference.

Lee, D. D., Seung, H. S., 1999. Learning the parts of objects by non-negative matrix factorization. Nature 401.

Ozerov, A., Févotte, C., Charbit, M., 2009. Factorial scaled hidden markov model for polyphonic audio representation and source separation. In: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA).

Rabiner, L. R., Juang, B. H., 1986. An introduction to hidden markov models. IEEE Acoustics, Speech and Signal Processing (ASSP) Magazine 3 (1), 4–16.

Raj, B., Smaragdis, P., 2005. Latent variable decomposition of spectrograms for single channel speaker separation. In: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA).

Smaragdis, P., Shashanka, M., Raj, B., 2009. A sparse non-parametric approach for single channel separation of known sounds. In: Neural Information Processing Systems (NIPS).

Virtanen, T., Cemgil, A. T., 2009. Mixtures of gamma priors for non-negative matrix factorization based speech separation. In: 8th International Conference on Independent Component Analysis and Signal Separation (ICA).