HIDDEN-ARTICULATOR MARKOV MODELS FOR SPEECH RECOGNITION

Matt Richardson, Jeff Bilmes and Chris Diorio

University of Washington
{mattr@cs, bilmes@ee, diorio@cs}.washington.edu

ABSTRACT

In traditional speech recognition using Hidden Markov Models (HMMs), each state represents an acoustic portion of a phoneme. We explore the concept of an articulator based HMM, where each state represents a particular articulatory configuration [Erler 1996]. In this paper, we present a novel articulatory feature mapping and a new technique for model initialization. In addition, we use diphone modeling which allows context dependent training of transition probabilities. Our goal is to confirm that articulatory knowledge can assist speech recognition. We demonstrate this by showing that our mapping of articulatory configurations to phonemes performs better than random mappings. Furthermore, we demonstrate the practicality of the model by showing that, in combination with a standard model, a 12-22% relative word error rate decrease occurs relative to the standard model alone.

1. INTRODUCTION

Hidden Markov Models (HMMs) are a popular approach for speech recognition. Commonly, a left-to-right Markov chain topology is used, where each phoneme is represented by a sequence of states (typically three) [Young 1996]. This “acoustic”-based model for speech recognition does not explicitly incorporate any knowledge of the source that produced the speech.

In contrast, we know that speech is formed by a human vocal tract, consisting of a number of articulators which shape and modify the sound in complex ways. We also know that this system of articulators is limited by physical constraints. This knowledge could allow us to construct a more realistic model of speech that might improve speech recognition. Such a model could have many advantages such as being better able to predict co-articulation effects, since they are due to physical limitations and energy-saving shortcuts in articulator movement [Hardcastle 1999]. Also, because articulatory configurations are shared across multiple phonetic conditions, the HAMM may need less training data.

The rest of the paper is as follows: Section 2 presents the model in detail and compares it with other work in the area of articulatory modeling. Section 3 describes how the model is initialized and trained, and Section 4 presents experimental results.

2. THE MODEL

The Hidden-Articulator Markov Model (HAMM) is based on the articulatory feature model presented in [Erler 1996]. In a HAMM, each articulator, *a*, can be in one of M_a positions. An *articulatory configuration* is an N-element vector $C=\{c_1,c_2,\ldots,c_N\}$, where c_a is an integer $0 \leq c_a < M_a$ and N is the number of articulators in the model.

A HAMM is simply an HMM in which each state represents an articulatory configuration. The state transition matrix is governed by dynamic constraints on articulator motion. Therefore, this model makes the assumption that the probability distribution of articulatory features is determined by the previous articulatory configuration, and is independent of any earlier configuration.

There are many potential advantages of a HAMM over the traditional HMM for speech recognition. The HAMM allows asynchrony between various articulators, which might more accurately model the production of speech [Deng 1994]. Also, the HAMM has prior knowledge about speech production, incorporated via its state space, transition matrices, and phoneme to articulator mappings. By using a representation that has a physical basis, it is easier to incorporate other knowledge such as co-articulation effects.

There has been much interest in incorporating articulatory knowledge into speech recognition. Work by Gupta and Schroeter [Gupta 1993] discusses the analysis-by-synthesis approach, which attempts to estimate the parameters of the Coker [Coker 1976] model, which is based on articulatory features. The analysis-by-synthesis work is often targeted toward speech compression, where the quality of the synthesis is more important than the accuracy of the estimated parameters.

A discussion of the inverse mapping problem, the mapping of acoustic features to articulatory configurations, can be found in [Bailly 1992]. One well-known difficulty with the inverse mapping problem is that many different articulatory configurations can produce a given sound; this is commonly referred to as the “many-to-one” problem.

In [Kirchhoff 1998] Kirchhoff demonstrates a system which uses artificial neural networks to estimate articulatory from acoustic features. When used in combination with an acoustic based HMM, the system achieves a lower word error rate in both clean and noisy speech.

Jaw	Lip Sep	Lip Round	Tongue Body	Tongue Body	Tongue Tip	Velic Aper.	Voicing
nearly closed	closed	rounded and or protruded	back	low and flat	low	closed	off
neutral	slightly apart	slightly rounded or tensed corners	slightly back	mid or central	neutral	open	on
slightly lowered	apart	neutral	neutral	mid-high	between or touching top teeth		
lowered	wide apart	wide	slightly forward	high	near alveolar ridge		
			forward		touching alveolar or upper ridge		

Figure 1: Articulatory feature space.

The HAMM can be cast as a factorial HMM [Saul 1999], with additional dependencies existing between separate Markov chains. Factorial HMMs have been applied to speech recognition [Logan 1998] but without the use of an articulatory feature space. We chose to implement the HAMM by constructing a constrained state space which is the Cartesian product of the components, as this allows us to use standard HMM algorithms for training and testing.

Phoneme Mapping

A word is a sequence of articulator targets. In mapping words to articulator configurations, we make the simplifying assumption that words can be modeled as a sequence of phonemes, each of which is mapped to a sequence of one or more articulatory configurations.

We used Edwards [Edwards 1997] as a guide to phonetics and speech production. Using this guide, we devised an articulatory feature space which is described by eight features (see Figure 1). For example, in our model we have quantized the separation of the lips into four possible positions, ranging from “closed” to “wide apart”.

Each phoneme’s articulatory characteristics were manually examined to determine the best mapping into our articulatory feature space. This mapping is given in the Appendix.

For some phonemes, an articulator may be in several possible locations. Phonemes are mapped into a vector of feature ranges; each feature can be in any of the values specified by the range. For example, when pronouncing the phoneme /h/, we allow a lip separation of either “apart” or “wide apart”, but do not allow the lips to be “closed” or “slightly apart”.

Some phonemes require a specification of articulator motion rather than static positioning. This occurs with the stops (/t/, /b/, etc..) and diphthongs (such as the “i” in “bite”). In these cases, a phoneme is produced by the movement from one articulatory state to another. Thus, we allow phonemes to be mapped to a sequence of articulatory configurations.

Static Constraints

We’ve also added static constraints, which limit the possible articulatory configurations:

(1) If the lips are widely separated then don't allow rounded or wide lip width.
(2) If the lips are closed then don't allow rounded or wide lip width.
(3) If the jaw is lowered, don't allow the lips to be closed or almost closed.
(4) If the tongue tip is near or is touching the alveolar ridge, then the tongue body must be mid-high or high, and the tongue body cannot be back or slightly back.
(5) If the velic aperture is open then voicing must be on.
(6) If the velic aperture is open then tongue cannot be forward or slightly forward.
(7) The velic aperture may only be open in a given articulatory configuration X if there is a transition directly from X to a nasal phoneme articulatory configuration.

Some of these constraints, such as (1), (3), and (4), are physical constraints, imposed by the limitations of the articulation system. Other constraints, such as (2), disallow states that are physically possible but would not normally be used while speaking naturally in American English. This set of static constraints reduces the number of states in the HAMM from what would be 25,600 to 6,676.

Dynamic Constraints

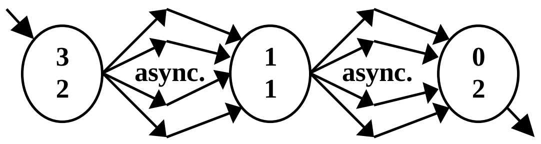
We impose dynamic constraints on the model to prevent physically impossible articulatory movement. We only allow the model to contain a transition from state C to state D if $\forall a$: $-1 \leq d_a - c_a \leq 1$, which imposes a continuity and maximum velocity constraint on the articulators.

Furthermore, we constrain the model so that the only allowable states, C, between any two target phoneme vectors, P and Q, are those which satisfy:

$$\forall a: \min(\{p_a, q_a\}) \leq c_a \leq \max(\{p_a, q_a\})^1$$

To construct a diphone, we list the sequence of articulatory targets from the first target of the first phoneme to the last target of the second phoneme (see Figure 2a). The

[1] Recall, p_a or q_a may be a set of values, see the section on phoneme mapping.



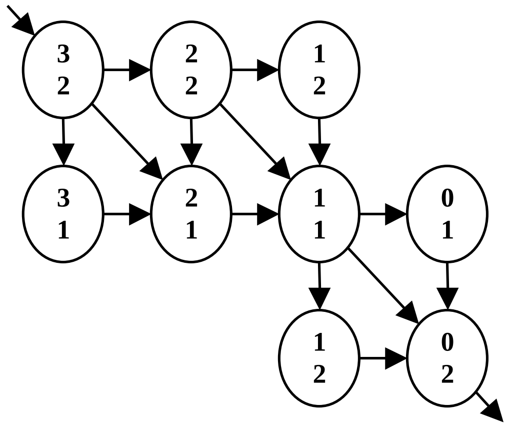


Figure 2a: (upper) A diphone model is a sequence of articulator configuration targets, with asynchronous articulatory movement in between.
Figure 2b: (lower) Example HMM transition graph for a diphone.

states between the targets are filled in and allowable transitions are added. For example, suppose N=2 and we are constructing a graph from phoneme P={ [3 2] → [1 1] } to Q={ [0 2] }. Then the resulting graph (assuming none of these states are removed by static constraints) is shown in Figure 2b.

Notice how the HAMM allows for asynchrony, whereby one articulator may move with or without other articulators moving, thus more accurately representing speech production. In addition, many different diphones may contain the same intermediate articulatory state, leading to a large amount of state sharing between diphones.

3. TRAINING

In this work, a HAMM is trained using the Baum-Welch algorithm. An HMM is constructed for each diphone using the above static and dynamic constraints. Words are constructed by concatenating diphone models. For instance, the model for the word "meatball" is the concatenation of the diphone models /m/-/i/, /i/-/t/, /t/-/b/, /b/-/a/, /a/-/l/. This allows the training to learn transition probabilities on a diphone level.

To reduce the model size, we removed states during training which had very low state occupation probabilities. After training, the number of parameters in the HAMM was reduced from 2 million to 591 thousand.

Initial Model Construction

Training requires an initial model, which is iteratively improved until it converges to a local optimum. The quality of the initial model can have a large effect on the performance of the trained model, and on its convergence. The states (articulatory configurations) in our model fall into two categories: (1) states which correspond to a phoneme, and (2) all other allowable states.

We use segmental k-means to determine an initial setting for the Gaussian parameters for states which fall into category (1) above. Each category (2) state is initialized by a weighted interpolation of the category (1) states. The weighting is given by the inverse Euclidean distance between the state being initialized, and the states from which we are interpolating (see Figure 3). We desire the parameters for the state being initialized to represent the probability distribution given by the following equation, where *S* is the set of all possible category (1) states, and w_i are inversely proportional to the Euclidean distance in our N-dimensional discrete articulatory feature space (where N=8 in our case):

$$p(x) = \sum_{i \in S} w_i N(x; \mu_i, \sigma_i) \quad \left(\sum_{i \in S} w_i = 1 \right)$$

The mean and variance of the above distribution is given by:

$$\hat{\mu} = \sum_{i \in S} w_i \mu_i \quad \hat{\sigma}^2 = \left[\sum_{i \in S} w_i \left(\sigma_i^2 + \mu_i^2 \right) \right] - \hat{\mu}^2$$

For category (2) states, we use a diagonal Gaussian with these means and variances. In the multi-component case, we do the same, using a random assignment of components from the states being interpolated to the component being initialized.

4. EXPERIMENTS AND RESULTS

Speech recognition results were obtained using PHONEBOOK, a large-vocabulary, phonetically-rich, isolated-word, telephone-speech database [Pitrelli 1995]. All data is represented using 12 MFCCs plus c_0 and deltas resulting in a 26 element feature vector sampled every 10ms. In the HAMM, each state uses a mixture of two diagonal covariance Gaussians.

Additionally, we define two models, *3state* and *4state*, which are standard left to right, diagonal Gaussian HMMs with 3 and 4 states per phoneme and with 16 and 24 mixtures per state respectively.

The training, development, and test sets are as defined in [Dupont 1997]. Test words do not occur in the training vocabulary, so test word models are constructed using diphones learned during training. Training was considered complete when the log-likelihood difference between successive iterations fell below 0.2%.

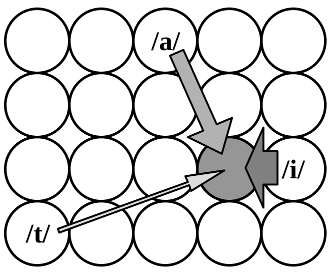


Figure 3: Sample state initialization. The shaded circle is a state being initialized. It is interpolated from states which are mapped to directly by a phoneme. The width of the arrow represents the weight given to each factor in the interpolation, which is proportional to the inverse Euclidean distance between them.

Lex. Size	75	150	300	600	params
original	3.23%	4.67%	6.69%	9.03%	522k
arbitrary	3.72 ± 0.08%	5.18 ± 0.06%	7.19 ± 0.20%	9.81 ± 0.22%	661k ± 10k
permutation	4.76 ± 0.24%	6.77 ± 0.40%	9.11 ± 0.43%	12.35 ± 0.35%	462k ± 13k

Table 1: Word Error Rate comparison of original phone mapping versus random mappings for various lexicon sizes. Random model results are given as mean ± standard error (we tested 5 arbitrary models and 2 permutation models). The original mapping is significantly better than either of the random mappings.

Lexicon Size	75	150	300	600	params
HAMM	3.23%	4.67%	6.69%	9.03%	522k
3state	1.88%	2.91%	4.20%	6.14%	105k
4state	1.45%	2.79%	4.04%	5.76%	203k
4state+3state	1.42%	2.49%	3.71%	5.46%	308k
4state+HAMM	1.27%	2.18%	3.29%	4.56%	725k

Table 2: Word Error Rate comparison showing the advantage of combining models. The best combination is the standard *4state* HMM with the HAMM.

	Normal		**Permutation**		**Arbitrary**	
Phone	Jaw	Nasal	Jaw	Nasal	Jaw	Nasal
a	0	1	1	0	0	0
b	2	0	0	1	1	1
c	1	0	0	0	2	1
d	0	0	2	0	1	0

Table 3: Sample phoneme mapping, highlighting the difference between *permutation* and *arbitrary* random mappings. *Permutation* is a reordering of the rows, while *arbitrary* is purely random. Notice how the *permutation* mapping retains the distribution of values for a given feature.

Comparison with Random

To verify that the HAMM uses the articulatory knowledge to its advantage, we compare its performance to that of a HAMM with no articulatory knowledge. To construct such a model, we use a random mapping of phonemes to articulatory features. To ensure a fair comparison, we use the same feature space, static, and dynamic constraints that were introduced in Section 2.

There are two ways to produce a random mapping. The first, referred to as *arbitrary*, simply selects a random value within the given feature range for all features across all phonemes. The second, referred to as *permutation*, randomly rearranges the original mapping. In other words, each phoneme is mapped in the same way as some randomly selected phoneme in the original mapping without duplication. Table 3 demonstrates the difference between the random mappings.

The arbitrary mapping is "more" random since it is drawing from a uniformly distributed state space. The permutation method produces a mapping that is still fairly random, yet retains the same distribution over features as the original mapping. For instance, in the original mapping, the *velic aperature* is open for only three phonemes. In a permutation mapping, this would still be the case, while in an arbitrary mapping, it would be open for approximately half of the phonemes.

Table 1 shows the results of this experiment on the test set. The arbitrary and permutation mappings both result in significantly worse ($p < 0.01$ using two-tailed z-test) word error rates than the original mapping. Furthermore, the arbitrary mapping requires significantly more parameters[2]. From these results, we conclude that the articulatory knowledge is indeed contributing to the performance of the HAMM.

Model Combination

The HAMM performs worse than the *3state* and *4state* models (see Table 2). We hypothesize that since it is based on articulatory knowledge, the HAMM makes different mistakes than the standard models, and thus a combination of them will result in improved performance.

There are a variety of techniques for combining models. One simple way is by a weighted sum of the models' log-likelihoods. The weighting of each model represents the prior confidence in its accuracy. If the model errors are independent, this will result in a higher accuracy [Bishop 1995].

In Table 2 we show the results of performing model combination. We give the *4state* model likelihoods a weight of 5.0 when combined with the HAMM (HAMM weight is 1), and give them a weight of 1.0 when combined with the *3state* model; these are the optimal weights based on the development set.

As can be seen from the table, the HAMM performs significantly worse than the *4state* model, but the combination of the two performs significantly better (12-22% relative decrease in WER versus *4state* alone). Also note that combining the *3state* model with the *4state* model has

[2] The arbitrary model begins with more parameters as well. In the arbitrary mapping, the beginning and ending phones of a diphone are more likely to contain different values for each feature since the entropy of each feature is higher than in the original or permuted mappings. This results in larger diphone models. Many of these states, however, were not removed by the state elimination algorithm, implying that they were being used by the model.

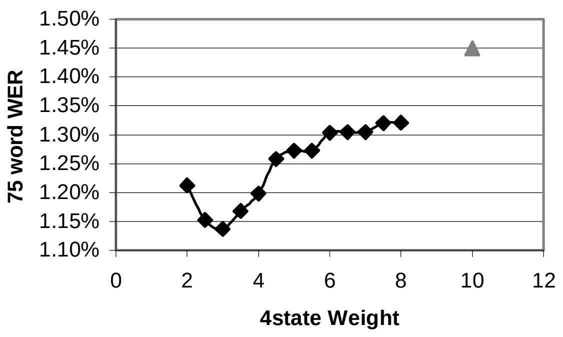


Figure 4: WER of the combined (*4state* + HAMM) model across various *4state* weightings for the 75 word lexicon test (the y-axis range has been scaled to improve resolution). The gray triangle in the upper right corner shows the performance of the *4state* model alone.

much less effect on the WER. This demonstrates the usefulness of an articulator-based model, in that it tends to make different mistakes than the standard acoustic models and thus can have practical use when used in combination with them.

Viterbi Path Through The Articulatory State Space

A Viterbi path decoding using our HAMM results in an estimation of articulatory feature values for an utterance. In Figure 5, we show a comparison of the spectrogram and the HAMM's automatically estimated articulatory features for the word "accumulation".

It is difficult to quantitatively compare the two figures. One feature which is easy to see in the spectrogram is voicing (feature 8), which seems to align very well with the HAMM's voicing feature. Another positive item to note is that the states evolve somewhat asynchronously, which is what we expect to find if the HAMM is indeed modeling the articulator movements [Deng 1994].

5. FUTURE WORK

We would like to extend this work by adding more articulatory knowledge, with rules for phoneme modification that arise as a result of physical limitations and shortcuts in speech production, as was done in [Erler 1996] (for example, vowel nasalization). Such rules may help speech recognition systems in the presence of strong coarticulation, such as in conversational speech.

We would also like to examine the possibility that HAMMs are more noise-robust than standard HMMs. We hypothesize that by having an articulatory basis, the HAMM is more attuned to the speech-like information contained in the utterance and thus is better equipped to ignore noise [Richardson 2000].

Finally, we believe that many aspects of the HAMM, such as model initialization, articulatory feature mapping, and constraints on articulator dynamics, could be improved by using a corpus which contains both speech and the articulator trajectories which produced the speech. One such corpus is under development by A. Wrench [Wrench 2000].

6. CONCLUSIONS

In this paper, we have introduced the hidden-articulator Markov model (HAMM) and have implemented it using HMMs. It does not perform as well as conventional HMMs, but we have demonstrated that the HAMM does indeed use articulator knowledge to its advantage. Furthermore, when combined with a traditional model, it achieves an overall WER reduction of 12-22% relative to the traditional model WER. These results demonstrate that a HAMM can have practical use in modern speech recognition systems.

ACKNOWLEDGEMENTS

We would like to thank Katrin Kirchhoff for her assistance with the speech tools and discussions about articulatory models. This work was supported by a Packard Foundation Fellowship.

REFERENCES

[1] G. Bailly, C. Abry, L.-J. Boe, R. Laboissiere, P. Perrier and J.-L. Schwartz (1992). "Inversion and Speech Recognition," Signal Processing VI: Proceedings of EUSIPCO-92, vol.1 pp.159-164

[2] C. Bishop (1995), *Neural Networks for Pattern Recognition.* (Oxford University Press)

[3] C. Coker (1976). "A model of articulatory dynamics and control," Proc. IEEE 64, 452-60.

[4] L. Deng and D. Sun (1994). "Phonetic Classification and Recognition Using HMM Representation of Overlapping Articulatory Features for all classes of English sounds," ICASSP, 1994, pp.45-8

[5] S. Dupont, H. Bourlard, O. Deroo, J.-M. Fontaine and J.-M. Boite (1997). "Hybrid HMM/ANN systems for training independent tasks: Experiments on phonebook and related improvements," ICASSP, 1997, pp.1767-70

[6] H.T. Edwards (1997), *Applied Phonetics: The Sounds of American English* second edition. (Singular, San Diego)

[7] K. Erler and G.H. Freeman (1996). "An HMM-based speech recognizer using overlapping articulatory features," J. Acoust. Soc. Am. 100, pp.2500-13

[8] S. Gupta and J. Schroeter (1993). "Pitch-synchronous frame-by-frame and segment-based articulatory analysis by synthesis," J. Acoust. Soc. Am. 94:5, Nov. 1993, pp.2517-30

[9] W.J. Hardcastle and N. Hewlett (eds.) (1999) *Coarticulation. Theory, Data and Techniques.* (Cambridge)

[10] K. Kirchhoff (1998). "Combining articulatory and acoustic information for speech recognition in noisy and reverberant environments", Proceedings of ICSLP, 1998, pp.891-4

[11] B. Logan and P. Moreno (1998). "Factorial HMMs for acoustic modeling," ICASSP, 1998 , pp.813-6

[12] J. Pitrelli, C. Fong, S.H. Wong, J.R. Spitz and H.C. Lueng (1995). "PhoneBook: A phonetically-rich iso-

lated-word telephone speech database" ICASSP, 1995 pp.101-4

[13] M. Richardson, J. Bilmes and C. Diorio (2000). "Hidden-Articulator Markov Models: Performance Improvements and Robustness to Noise," ICSLP 2000, to appear.

[14] L. Saul and M. Jordan (1999). "Mixed Memory Markov models: decomposing complex stochastic processes as mixtures of simpler ones," Machine-Learning 37:1, Oct 1999, pp.75-87

[15] A. Wrench (2000). "A Multichannel/Multispeaker Articulatory Database for Continuous Speech Recognition Research," Workshop on Phonetics and Phonology in ASR, Saarbruecken, Germany, 2000.

[16] S. Young. "A Review of Large-vocabulary Continuous-speech Recognition," IEEE Signal Processing Magazine, 13:5, Sept. 1996, pp.45-57

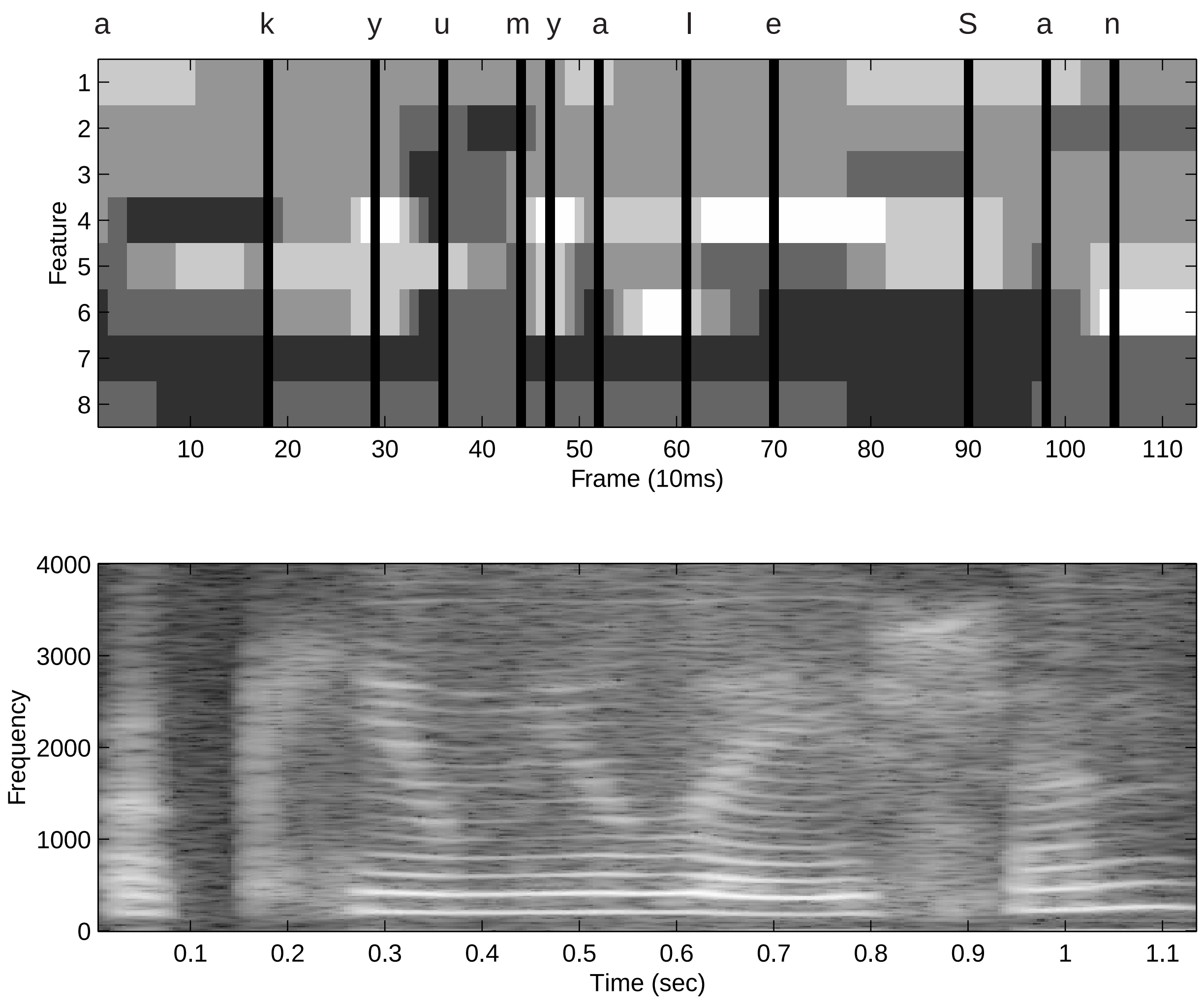


Figure 5: HAMM Viterbi path decoding for the word "accumulation." The lower half of the figure is a spectrogram of the speech. The upper half shows the estimated articulatory configurations over time (note: features are numbered 1-8 with 1=Jaw and 8=Voicing). The black vertical lines denote the estimated diphone boundaries.

APPENDIX

Below is the mapping from phonemes to articulatory configurations. Note that some phonemes have multiple values for a given feature, such as the tongue tip position in phoneme /R/. Some phonemes also are defined as a sequence of configurations, such as the phoneme /p/, which is formed by bringing the lips together (lip separation=0, "closed") to temporarily stop the flow of air, and then separating them (lip separation=2, "apart").

phoneme	sample word	jaw	lip separation	lip width	tongue body (back/fwd.)	tongue body (low/high)	tongue tip	velic aper.	voiced
i	b**EA**t	0	1	2	4	3	0	0	1
I	b**I**t	3	2	2	4	2	0	0	1
e	b**AI**t	1	2	2	4	1	0	0	1
E	b**E**t	3	2	2	4	1	0	0	1
@	b**A**t	3	3	1	3	0	0	0	1
a	b**O**b	3	2	2	2	0	0	0	1
c	b**OUGH**t	3	2	0	1-2	3	0	0	1
o	b**OA**t	3	2	0	1	1	0	0	1
^	b**U**t	2	2	2	2	1	0	0	1
u	b**OO**t	1	1	0	0	3	0	0	1
U	b**OO**k	1	2	1	0	3	0	0	1
Y	b**I**te								
	onset	3	2	2	3	0	0	0	1
	offset	1-2	2	2	4	3	0	0	1
O	b**OY**								
	onset	2	2	0	1	0-1	0-1	0	1
	offset	0-1	2	1-2	4	3	1	0	1
W	b**OU**t								
	onset	3	2	2	3	0	0	0	1
	offset	1-2	2	0	0	3	0	0	1
R	b**IR**d	2	2	0	2-3	2	0-1	0	1
x	sof**A**	2	2	2	2	1	0	0	1
X	butt**ER**	2	2	1	2	2	0-1	0	1
l	**L**et	1	2	2	3	2	4	0	1
w	**W**et	1	2	0	0	3	1	0	1
r	**R**ed	1	2	1	2	2	3	0	1
y	**Y**et	1	2	2	4	3	3	0	1
n	**N**eat	1	1	2	2	3	4	1	1
m	**M**eet	1	0	2	2	1	1	1	1
G	si**NG**	1	2	2	0	3	1	1	1
h	**H**eat	2	2-3	2	2	1	1	0	0

phoneme	sample word	jaw	lip separation	lip width	tongue body (back/fwd.)	tongue body (low/high)	tongue tip	velic aper.	voiced
s	**S**ee	1	2	1-2	3	2-3	0-1	0	0
S	**S**he	2	2	1-2	3	3	0	0	0
f	**F**ee	2	0	2	2	1	1	0	0
T	**Th**igh	2	2	2	4	2	2	0	0
z	**Z**oo	1	2	1-2	3	3	0-1	0	1
Z	mea**S**ure	2	2	1-2	3	3	0	0	1
v	**V**an	2	0	2	2	1	1	0	1
D	**Th**y	2	2	2	4	0	2	0	1
p	**P**ea								
	setup	1	0	2	2	1	1	0	0
	release	1	2	2	2	1	1	0	0
t	**T**ea								
	setup	1	1	2	4	3	4	0	0
	release	1	2	2	4	2	3	0	0
k	**K**ey								
	setup	1	2	2	0	3	1	0	0
	release	1	2	2	0	2	1	0	0
b	**B**ee								
	setup	1	0	2	2	1	1	0	1
	release	1	2	2	2	1	1	0	1
d	**D**ay								
	setup	1	1	2	4	3	4	0	1
	release	1	2	2	4	2	3	0	1
g	**G**eese								
	setup	1	2	2	0	3	1	0	1
	release	1	2	2	0	2	1	0	1
C	**Ch**ur**CH**								
	start	2	2	1-2	4	3	4	0	0
	end	1	2	2	3	3	0	0	0
J	**J**u**DG**e								
	start	2	2	1-2	4	3	4	0	1
	end	1	2	2	3	3	0	0	1