Articulatory Feature-based Methods for Acoustic and Audio-Visual Speech Recognition: 2006 JHU Summer Workshop Final Report[1]

Karen Livescu (klivescu@csail.mit.edu),
Özgür Çetin (ocetin@icsi.berkeley.edu),
Mark Hasegawa-Johnson (jhasegaw@uiuc.edu),
Simon King (simon.king@ed.ac.uk),
Chris Bartels (bartels@ee.washington.edu),
Nash Borges (nashborges@jhu.edu),
Arthur Kantor (akantor@uiuc.edu),
Partha Lal (partha.lal@gmail.com),
Lisa Yung (lyung1@jhu.edu),
Ari Bezman (ari.bezman@dartmouth.edu),
Stephen Dawson-Haggerty (sdawson@fas.harvard.edu),
Bronwyn Woods (bwoods1@swarthmore.edu)

[1]Edited portions of this report have appeared or will appear in references [56, 55, 15, 26, 40]. We gratefully acknowledge a number of remote collaborators that contributed to this project before and during the workshop: Xuemin Chi, Joe Frankel, Lisa Lavoie, Mathew Magimai-Doss, and Kate Saenko. In addition, we are grateful for assistance and support from Jeff Bilmes, Nancy Chen, Trevor Darrell, Edward Flemming, Eric Fosler-Lussier, Jim Glass, Katrin Kirchhoff, Daryush Mehta, Stefanie Shattuck-Hufnagel, Janet Slifka, Amar Subramanya, Fred Jelinek, Sanjeev Khudanpur, the staff and faculty of CLSP, the National Science Foundation, DARPA, and the Department of Defense. Phonetic alignments for some experiments were provided by SRI, and additional computational infrastructure was generously provided by CSTR, U. Edinburgh; IFP, UIUC; the SSLI lab, U. Washington; ICSI Berkeley; and CSAIL, MIT.

Abstract

We report on investigations, conducted at the 2006 JHU Summer Workshop, of the use of articulatory features in automatic speech recognition. We explore the use of articulatory features for both observation and pronunciation modeling, and for both audio-only and audio-visual speech recognition. In the area of observation modeling, we use the outputs of a set of multilayer perceptron articulatory feature classifiers (1) directly, in an extension of hybrid HMM/ANN models, and (2) as part of the observation vector in a standard Gaussian mixture-based model, an extension of the now popular "tandem" approach. In the area of pronunciation modeling, we explore models consisting of multiple hidden streams of states, each corresponding to a different articulatory feature and having soft synchrony constraints, for both audio-only and audio-visual speech recognition. Our models are implemented as dynamic Bayesian networks, and our experiments have been performed on two types of tasks: For audio-only recognition, we use tasks from the Small-Vocabulary Switchboard (SVitchboard) database; and for audio-visual recognition, we use a subset of the CUAVE read digits database. Finally, we have collected a small set of manual transcriptions at the articulatory feature level. We describe the data collection effort and analyze articulatory feature classification and alignment performance using the manual labels as reference.

Contents

Chapter 1

Introduction

Articulatory features (AFs) have a long history in proposals for automatic speech recognition (ASR) architectures (e.g., [74, 70, 21]). These are usually motivated by the intuitions that (1) models related to speech production should better account for coarticulation effects, (2) certain aspects of articulation are more acoustically salient or robust to noise than others, and (3) models based on several classifiers with a small number of classes each (i.e. articulatory feature classifiers) may make better use of sparse training data than ones based on a single classifier with a large number of classes (phone or phone-in-context classifiers). Articulatory approaches may also be more applicable to cross-domain or cross-linguistic adaptation (see, e.g., [82]) because of their potential for more efficient use of training data, as well as the greater sharing of feature inventories than of phone inventories across languages. Another application is audio-visual speech recognition (AVSR), where articulatory features provide a natural explanation for the observed asynchrony between the audio and video signals. AFs have been proposed as the basis for an AVSR architecture [67], and have been used for audio-visual phonetic recognition [53], but to our knowledge have not been used in a complete continuous speech recognizer. Finally, articulatory approaches using multiple streams of states for the different features are more compatible with phonological theories such as autosegmental [32] and articulatory phonology [12].

Approaches based on articulatory features have had some success in recent years. Recognition improvements have been obtained using AF classifiers in combination with a phone-based recognizer, for example in noisy conditions [49] or for varying speaking styles [79]. In the area of pronunciation modeling, improvements have been obtained in lexical access experiments using multistream models that account for articulatory asynchrony and reduction [58, 54].

The ultimate goal of our work is to build complete continuous speech recognizers that use AFs at all levels. This includes (1) multiple, loosely-coupled feature-specific hidden state streams that can account for the semi-independent behavior of the articulators, and (2) observations and/or observation models tailored to these multiple streams. In order to clarify our terminology, we begin with a few definitions. We assume for the current work that the task is first-pass Viterbi decoding, in other words finding the jointly most likely string of words w^* and set of state variable assignments q^*:

$$\{w^*, q^*\} = \arg\max_{w,q} p(o|q)p(q|w)p(w), \tag{1.1}$$

where o are the observation vectors and we have made the standard assumption that the observations are independent of the words w given the states q. In hidden Markov model (HMM)-based recognizers, q is the sequence of hidden state values. In our case, q is not necessarily a single stream of states but any collection of hidden variables corresponding to the sub-word representation. For example, q may be a collection of multiple hidden state streams. We refer to $p(o|q)$ as the *observation model*, and to $p(q|w)$ as the *pronunciation model*. We note that this is a non-standard definition of the pronunciation model and refers to the entire probabilistic mapping between words and sub-word structure. $p(w)$ is the language model, which we assume to be fixed and do not discuss in the remainder of this report. We further assume that the observations are frame-based, i.e. that we have one observation vector per unit of time. This is in contrast to previous related work using AFs in a landmark-based, rather than frame-based, architecture [38, 39].

1.1 Design choices for AF-based recognition

Within the broad framework we have described, there are a number of design choices to be made, which together define a large space of possible models. Figure 1.1 describes some of the main dimensions along which AF-based models may differ. Some of these have been explored to various extents while others have not.

The main design choices depicted in Figure 1.1 are whether or not the hidden state is factored into multiple state streams (presumably one per feature or group of features), and whether or not the observation model is factored; in other words, whether $p(q|w)$ or $p(o|q)$ are factored. For example, in [21, 73], the hidden state is unfactored but is constructed as a combination of articulatory feature states, and the observation model is unfactored as well. In [61, 49, 44], the hidden state is the standard phonetic HMM state, but the observation model is factored according to the feature values of the current phone state, with varying types of classifiers (Gaussian mixture, neural network, and support vector machine) used for each stream's observation model.

When the hidden state is factored, i.e. $q = \{q_1, \ldots, q_F\}$ (where F is the number of feature state streams), then there is a variety of choices of mechanisms for controlling the asynchrony between the state streams. The streams could be completely free to desynchronize within some unit (such as a phone or syllable); the observation model may then be factored, $p(o|q_1, \ldots, q_F) = \prod_{i=1}^{F} p(o_i|q_i)$, as in [47, 85], or unfactored, as in a factored HMM [28]. The asynchrony may be controlled by coupling the state transition probabilities, as in coupled HMMs [68, 64]. Alternatively, "soft synchronization" constraints may be imposed throughout a larger unit such as a word, as in [58, 54, 39], or even across word boundaries (which, to our knowledge, has not been attempted).

In all cases, the observation model may be context-independent or context-dependent, taking into account the neighboring phonetic context. Finally, there are some very important additional design choices not depicted in Figure 1.1, such as the choice of phonetic and articulatory state inventories.

In this report, we describe investigations into several approaches for observation and pronunciation modeling, covering parts of the taxonomy of Figure 1.1. For observation modeling, we experiment with two approaches using multilayer perceptron (MLP) classifiers of AFs: a factored "hybrid" approach, in which the MLP outputs are used directly as estimates of $p(o|q_i)$, inspired by phone-based hybrid approaches [62] and similar to one of the approaches in [48]; and both factored and unfactored "tandem" approaches, in which the log MLP outputs are concatenated to a standard acoustic vector after undergoing dimensionality reduction [91]. We investigate "embedded training" of the MLPs, in which a set of training data is realigned using one of the new models and the MLPs are retrained [85].

For pronunciation modeling, we experiment with a factored model consisting of multiple hidden streams of states, each corresponding to a different articulatory feature and having soft synchrony constraints, for both audio-only and audio-visual speech recognition. We use models similar to those of [58] and, for audio-visual recognition, ones based on coupled HMMs.

In the current work, we do not combine the new observation and pronunciation models; for observation modeling experiments, we assume that the hidden states are a single stream of phonetic units, and for pronunciation modeling we use a Gaussian mixture-based observation model. In all cases, we represent and implement the models as dynamic Bayesian networks (DBNs) [63]. As part of this work, we have also collected a new set of manual transcriptions at the articulatory feature level, and we present a set of observations that can be made by comparing AF classification and alignment performance against the manual alignments.

1.2 Goals and contributions

The initial, broad goals of this project were to explore articulatory models in a unified approach, in which both multiple hidden articulatory streams and feature-specific observation models could be combined in a principled way. Dynamic Bayesian networks provide a good architecture for this work, as they allow us to naturally represent different model variants in a uniform way.

For the 2006 Workshop, we divided this long-term goal into several narrower goals, based on dividing the problem into pronunciation modeling and observation modeling work. We studied pronunciation and observation models separately, while maintaining a DBN-based framework that would allow us to eventually

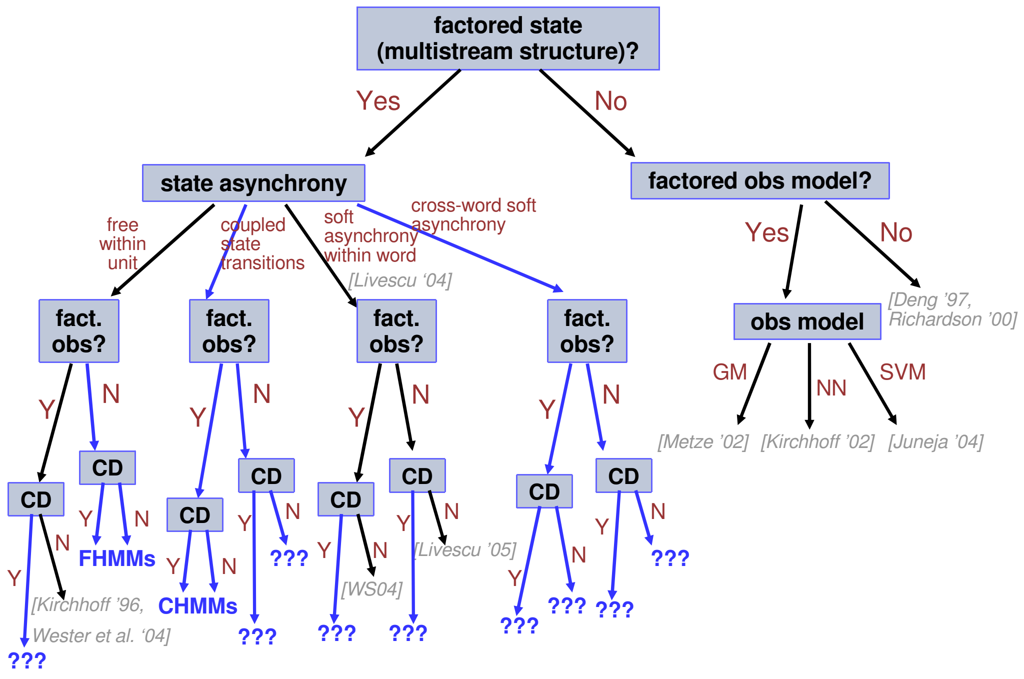

factored state (multistream structure)?
Yes
No
state asynchrony
factored obs model?
free within unit
coupled state transitions
soft asynchrony within word
cross-word soft asynchrony
[Livescu '04]
Yes
No
[Deng '97, Richardson '00]
fact. obs?
obs model
GM
NN
SVM
Y
N
CD
[Metze '02]
[Kirchhoff '02]
[Juneja '04]
FHMMs
CHMMs
???
[Kirchhoff '96, Wester et al. '04]
[WS04]
[Livescu '05]

Figure 1.1: A taxonomy of design choices in articulatory feature-based models

combine the most promising ideas from both components. The specific goals we focused on are:

1 *Pronunciation modeling using multiple, hidden, asynchronous articulatory feature streams.* For this work, we used Gaussian mixture observation models, where the acoustic observations depend on *all* of the hidden streams, similarly to factorial HMMs [28]. The bulk of this work involved models allowing context-independent asynchrony within word boundaries; we also began investigating ways of allowing asynchrony across word boundaries, modeling features that do not achieve their target values, and conditioning the asynchrony on certain types of context.

2 *Observation modeling using MLP classifiers of articulatory features*, in both *hybrid* and *tandem* settings. For this work, we used a standard phone-based pronunciation dictionary. We also investigated re-training of the MLPs using forced alignments of training data generated by articulatory recognizers.

3 *Application of articulatory approaches to audio-visual speech recognition*, using multi-stream models based on the ones developed for the pronunciation modeling work.

4 *Analysis of MLP classifier accuracy and articulatory forced alignment accuracy*, measured against human-labeled data. No prior work, to our knowledge, has compared automatically generated feature labels to human decisions.

The main contributions of this project are progress made in all of these areas. For both pronunciation modeling and observation modeling, we have implemented a range of model variants and studied their behavior in training and testing. The main results are: (1) The AF tandem-based systems achieve improvements over HMM baselines, and benefit from factoring the observation model; (2) the hybrid systems remain worse than the baseline models, except when combining hybrid and standard Gaussian mixture observation models; (3) the multistream pronunciation models are close to, but still worse than, single-stream phone-based models for audio-only recognition; (4) for AVSR, the multistream models match the performance of phoneme-viseme models; (5) AF classification and alignment are improved after re-aligning the training data using several of the new models and re-training the classifiers. Surprisingly, we find that some of the best classification and alignment improvements are obtained with the worst-performing multistream models, suggesting a possible future use for such models.

Our goals were also supported by several secondary tasks:

1 *Definition of articulatory feature sets.* Although there has been previous work on articulatory feature-based approaches to ASR, there has not been a great deal of effort invested in designing feature sets for use in describing conversational speech and with certain desirable properties, such as the ability to describe various partial changes.

2 *Collection of human-labeled data* for the analysis described above. This consists of a multi-tiered manual transcription of a small amount of data, insufficient to train models but sufficient to test classifier and alignment accuracy.

3 *Infrastructure* for generalized clustering of DBN states, analysis and debugging of DBN models, and site-independent distributed experiments using the Graphical Models Toolkit [30].

These tasks, while secondary to the main goals, are necessary for progress in AF-based ASR and will hopefully be helpful for future work in this area.

The main sources of data for this project were SVitchboard, a set of small-vocabulary subsets of Switchboard, for audio-only experiments; CUAVE, a database of spoken digits for audio-visual experiments; and the newly collected set of human-labeled utterances for analysis purposes.

1.3 Outline of this report

The remainder of this chapter briefly introduces dynamic Bayesian networks and their application to speech recognition. Chapter 2 describes the articulatory feature sets, data sources, and baselines for our experiments, as well as several tools developed for this project. The following three chapters describe the main experimental work on pronunciation modeling (Chapter 3) and observation modeling (Chapter 4), as well as analysis of the MLP classifiers and forced alignments (Chapter 5). Chapter 6 summarizes our main findings and discusses ongoing and future work.

1.4 Dynamic Bayesian networks

All of our models are implemented as dynamic Bayesian networks (DBNs), a type of graphical model. Graphical models are a framework for representing a joint distribution over a set of variables via a graph, where each node in the graph corresponds to a variable and the graph edges encode the independence properties among the variables. The joint distribution of the variables is given as a product of functions, each corresponding to a node or a set of nodes. Extremely general algorithms exist for performing graphical model inference–answering questions of the form "given the values of the variables in set A, what is the distribution of the variables in set B?"–which is a subroutine of decoding and maximum-likelihood training. This allows one to experiment with large classes of models without the need to develop new training and decoding algorithms for each variation.

Bayesian networks are a type of graphical model in which the graph is a directed acyclic graph, and each node is associated with its conditional distribution given its parents in the graph. The joint probability of all of the variables $x_1, \ldots, x_N$ is given by the product of the local conditional probabilities:

$$p(x_1, \ldots, x_N) = \prod_{i=1}^{N} p(x_i | pa(x_i)), \tag{1.2}$$

where $pa(x_i)$ is the set of parents of variable x_i.

A dynamic Bayesian network (DBN) is a Bayesian network with a repeating structure, which is useful for modeling stochastic processes over time or space. The repeating portion of the structure is referred to as a *frame*. An example of a DBN is a hidden Markov model (HMM), in which each frame consists of a state variable and an observation variable. The local conditional probabilities associated with the nodes are the transition probabilities $p(q_{t+1}|q_t)$ for the state variable and the emission probabilities $p(o_t|q_t)$ for the observation variable. Figure 1.2 shows the representation of an HMM as a DBN.

Here and throughout, we use the following conventions in DBN figures:

- Square and circular nodes correspond to discrete and continuous variables, respectively. Shaded nodes are observed. Nodes with thick edges correspond to deterministic variables.
- Dotted edges correspond to *switching* parents. A switching parent determines which of the other (non-switching) parents, or which conditional distribution, are used.
- In general, we show two frames of each DBN and the edges between them (the models we use do not require edges that extend further than this). For ease of viewing, we do not explicitly show details of the initial and final frame that may differ from intermediate frames.

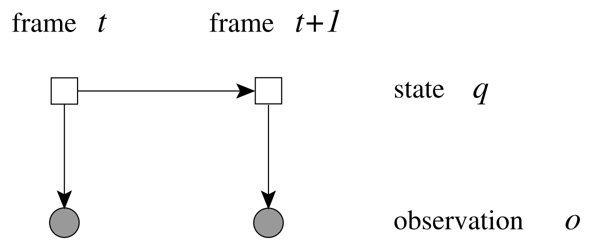


Figure 1.2: The DBN corresponding to an HMM. See the text for notational conventions.

DBNs have recently been gaining popularity in speech recognition research. Zweig [92] developed the first implementation of a speech recognizer as a DBN, showing how a standard HMM-based speech recognizer can be represented as a DBN and how it can be extended with the use of additional "auxiliary" variables. Since then DBNs have been used for speech recognition in various ways. For example, Bilmes introduced buried Markov models [8], in which dependencies between the observations in successive time frames are learned and added to an HMM-based structure. Several investigations have used an HMM-based structure with one or two additional auxiliary variables encoding articulatory features, pitch, or speaking rate [92, 80].

Finally, DBNs have been used to represent multiple acoustic observation streams [89] or simultaneous audio and video streams for audio-visual speech recognition [64, 33]. Research in speech recognition using DBNs has been facilitated by the development of the Graphical Models Toolkit (GMTK) [10], which we use for all of our experiments. Bilmes and Bartels [9] provide a survey of graphical model structures for ASR. The DBN models we use here for pronunciation modeling are based on those of Livescu and Glass [54, 57, 58].

Figure 1.3 shows a DBN, based on [92], that implements a standard phone HMM-based decoder with a bigram language model, which serves as our baseline model for all experiments. The variables in this model are:

- $word$: The current word.
- $wordTransition$: A binary variable, with value 1 if the current frame ends the current word. It serves as a switching parent to $word$: If $wordTransition = 1$, then the following $word$ is drawn from the bigram LM distribution; otherwise, the following $word$ is copied from the current frame.
- $subWordState$: A counter indexing how far along we are in the current word. For example, if a word contains ten phones with three states each, then $subWordState$ takes on the values $\{0, 1, \ldots, 29\}$ as we proceed through the word.
- $phoneState$: The current phone state ([aa0], [aa1], [aa2], [s0], [s1], [s2], ...). It is given deterministically by the current $word$ and $subWordState$.
- $stateTransition$: A binary variable, with value 1 if the current frame ends the current phone state. It depends on $phoneState$ and encodes the HMM transition probabilities.
- obs: The acoustic observation vector.

Note that the same structure can be used to implement either a monophone or a within-word triphone recognizer, depending on the set of values that $phoneState$ can take on. We have omitted, for ease of viewing, a variable encoding the pronunciation variant, which we use in our experiments to allow for a dictionary with multiple variants per word. This variable is an additional parent to $phoneState$ and $wordTransition$.

The recognizer can be trained via Expectation-Maximization (EM) [20] as is standard in ASR. If word alignments are available in training, then the same DBN can be used for training and decoding, with the exception that the $word$ variable is observed in training. If word alignments are not available, a slightly different structure is needed to account for the known word transcription. Such a structure is shown in Figure 1.4. Here the variable $wordCounter$ is initialized to 0 in the first frame and incremented whenever there is a word transition; the $word$ variable now depends only on $wordCounter$.

In the pronunciation modeling experiments of Chapter 3, we replace the sub-word state structure with a similar structure for each AF stream. In the observation modeling experiments, we use the baseline structure as is, but with the addition of phone-dependent AF variables (for hybrid systems) or replace the observation variable with one or more tandem feature vectors.

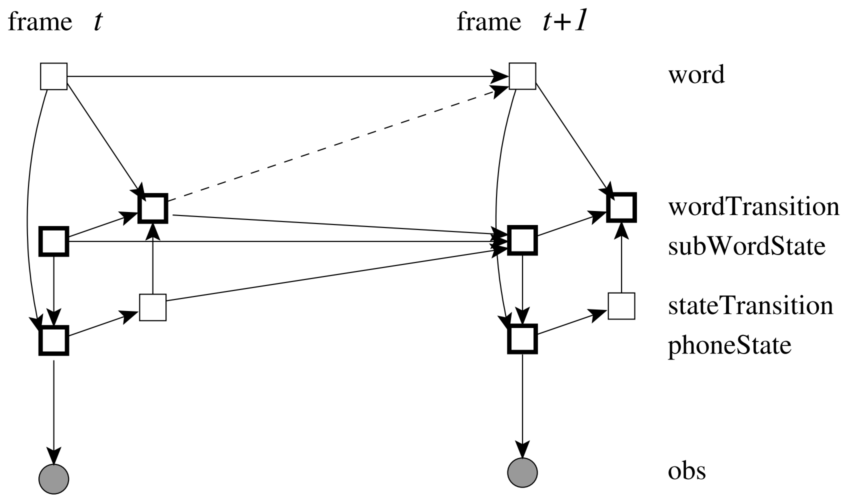


Figure 1.3: A monophone/within-word triphone HMM-based DBN for decoding.

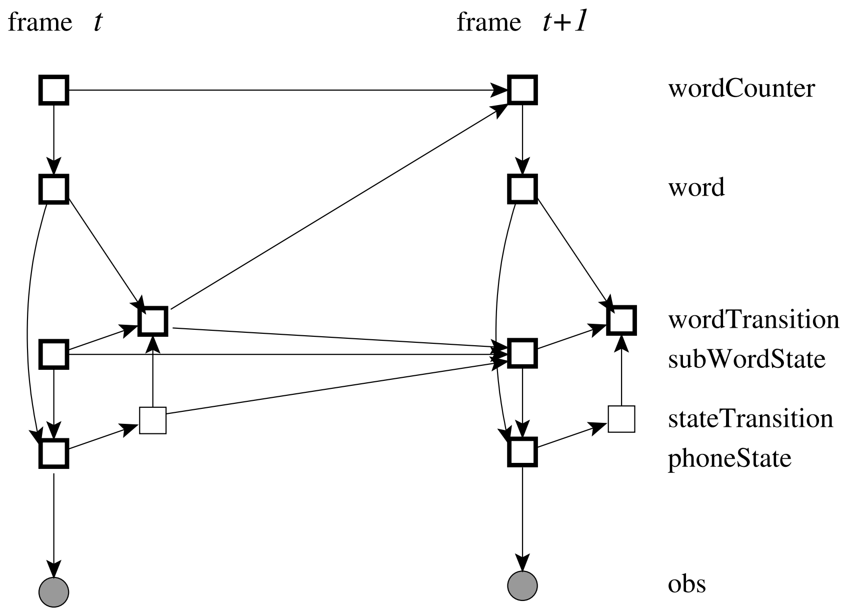


Figure 1.4: A monophone/within-word triphone HMM-based DBN for training without word alignments.

Chapter 2

Feature sets, data, baselines, and infrastructure

This chapter provides information on several unrelated background topics: the articulatory feature sets we developed and used for this project; the main data sources and baselines used in our experiments; and tools developed for the workshop.

2.1 Articulatory feature sets

This section describes the articulatory feature sets used in the project. This includes two kinds of feature sets: One based on articulatory phonology [12] and prior work on feature-based pronunciation modeling [54], intended for our pronunciation modeling experiments; and another more closely related to the dimensions used in the International Phonetic Alphabet chart [2], intended for observation modeling.

The rationale behind the use of two different feature sets is that different type of features better satisfy the assumptions we would like to make about pronunciation or observation models. In particular, for observation models based on feature-specific classifiers, we would like each feature to be independently inferrable from the acoustics. In contract, for pronunciation models using a separate hidden state sequence for each feature, we would like the features to be independent of each other. For example, *place of articulation* and *manner of articulation* have direct acoustic correlates, and are therefore appropriate choices for feature-specific classifiers; but they are not independent (e.g. if *manner* = "vowel", then most places of articulation are impossible). On the other hand, the positions of the lips and tongue are approximately independent, making it reasonable to represent them as separate streams in a multi-stream pronunciation model; however, the tongue and lip positions are not independent given the acoustics so should not be classified separately. This can be expressed in terms of conditional independence properties as follows: If q_M, q_P, q_L, q_T are the states of the manner, place, lip position, and tongue tip position features, respectively, then:

$$q_M \perp q_P | o, \quad q_M \not\perp q_P \tag{2.1}$$

$$q_L \not\perp q_T | o, \quad q_L \perp q_T, \tag{2.2}$$

where $\perp$ denotes "is independent of" and o are the acoustic observations.

Given that we would like to use different feature sets for different parts of a recognizer, how can the two be combined in a single recognizer? At least one way was developed by Hasegawa-Johnson *et al.* at the 2004 Johns Hopkins summer workshop [38, 39]. In this approach, which used a Bayesian network formulation, the two feature sets are explicitly represented as separate variables, with one set being a deterministic mapping of the other (e.g. "*manner* = *vowel* if *lip constriction* > *narrow* and *tongue tip constriction* > *narrow* and *tongue body constriction* > *narrow* and *glottis* ≠ *stop*"). As long as such a deterministic mapping exists, the use of the two feature sets is fairly straightforward.

In this project, however, we experimented separately with pronunciation models and with observation models, and used the corresponding feature set for each type of model. Only in some of the analysis work, where we compare the alignment performance of different models, do we combine the feature sets (see Chapter 5).

2.1.1 Articulatory phonology-based feature set for pronunciation modeling

For pronunciation modeling, we used a feature set based on articulatory phonology [12] and previously used by [54]. The theory of articulatory phonology provides a relatively precise representation of the types of asynchrony observed, in practice, among loosely coupled articulators. It has been developed in part through the observation of measured articulatory trajectories. In the theory of articulatory phonology, the entry for each word in the mental lexicon is composed of loosely ordered, discrete-valued "gestures." Each gesture defines the target for one or more "tract variables", such as locations and degrees of lip opening, tongue tip, and tongue body, and the states of the glottis and velum. In normal speech production, all of the gestures in a word are always produced; there is no such thing as gesture deletion, substitution, or insertion. The wide range of pronunciation variability observed in real-world pronunciation is accounted for by gesture overlap or by a vocal tract variable's failure to reach its target.

Our pronunciation modeling work uses ideas similar to those of articulatory phonology, and the feature set is correspondingly based on the vocal tract variables. We started with a set, shown in Table 2.1, in which each feature corresponds to one of the vocal tract variables. We did not use this feature set directly in our models, however. We make the assumption that LIP-LOC and LIP-OPEN are always synchronized; the four tongue features are always synchronized; GLOT and VEL are always synchronized; and there are no substitutions of feature values from the dictionary to the surface pronunciation. In this case, we can collapse each set of synchronous features into a single feature, whose state space of possible values may be much smaller than the product space of the component features.

The resulting feature set, which was actually used in experiments, is shown in Table 2.2. We refer to this set as the "AP (articulatory phonology) feature set". Appendix A gives the mapping we used from phones to feature values. In the audio-visual recognition experiments, the feature set was slightly modified; see Section 3.4 for more details.

2.1.2 IPA-inspired feature set for observation modeling

In developing a feature set for observation modeling, our goal was to be able to represent all of the articulatory configurations allowed in the AP feature set, while having each feature be independently discriminable from the acoustics.

Our starting point was the dimensions used to define phonetic symbols in the IPA [2]: manner, place, nasality, voicing, rounding, vowel height, and vowel frontness. We modified the feature set as needed in the process of developing a set of manual transcriptions. For example, we added "irregular voicing" and "voicing + aspiration" as values of a glottal state feature.

This feature set was intended to be used as both (1) the output classes of the neural network articulatory feature classifiers, and (2) the label inventory for manual transcriptions. In practice, there were some differences in the feature sets used for these two purposes. First, in the initial classifier training labels, only one place and manner occur at any one time. Second, some feature values may be acoustically discriminable (and therefore used in the manual transcriptions) but do not exist in the initial classifier training labels; this applies, for example, to irregular voicing and to some of the vowel classes.[1] Finally, we found it useful to have a special "silence" value for all feature classifiers, but not necessarily for manual labeling.

Table 2.3 defines the feature set. Feature values in italics were used in manual labeling but not in the AF classifiers. In addition, as previously mentioned, all classifiers used an additional "silence" class. Appendix A provides the phone-to-AF mapping for this feature set, as well as the detailed instructions used in the manual labeling.

A few aspects of this feature set are noteworthy. First, the consonant place and degree features (*pl1, dg1, pl2, dg2*) are intended to allow for up to two simultaneous constrictions, as might occur in /p r/ or /p l/ clusters, where the rhotic/lateral constriction may overlap with the stop burst. *Pl1* and *dg1* are used for

[1]This could be amended by generating training labels using forced alignments output by a generative AF-based model rather than a phone-based model. We experimented with AF-based forced alignments (see Section 5.3), but did not use them to train classifiers.

Feature	Description	# values	value = meaning

LIP-LOC	position (roughly, horizontal displacement) of the lips	3	P = protruded (rounded)
			L = labial (default/neutral position)
			D = dental (labio-dental position)

LIP-OPEN	degree of opening of the lips	4	CL = closed
			CR = critical (labial/labio-dental fricative)
			N = narrow (e.g., [w], [uw])
			W = wide (all other sounds)

TT-LOC	location of the tongue tip	4	D = inter-dental (e.g., [th], [dh])
			A = alveolar (e.g., [t], [n])
			PA = palato-alveolar (e.g., [sh])
			R = retroflex (e.g., [r])

TT-OPEN	degree of opening of the tongue tip	6	CL = closed (stop consonant)
			CR = critical (fricative, e.g. [s])
			N = narrow (e.g. [r] or alveolar glide)
			MN = medium-narrow
			M = medium
			W = wide

TB-LOC	location of the tongue body	4	PA = palatal (e.g. [sh], [y])
			V = velar (e.g., [k], [ng])
			U = uvular (default/neutral position)
			PH = pharyngeal (e.g. [aa])

TB-OPEN	degree of opening of the tongue body	6	CL = closed (stop consonant)
			CR = critical (fricative, e.g. [g] burst)
			N = narrow (e.g. [y])
			MN = medium-narrow
			M = medium
			W = wide

VEL	state of the velum	2	C = closed (non-nasal)
			O = open (nasal)

GLOT	state of the glottis	3	CL/ST = closed (glottal stop) CR/VO = critical (voiced) O/VL = open (voiceless)

Table 2.1: Definition of a feature set based on the vocal tract variables of articulatory phonology [12] and previously used by [54].

Feature	Description	# values	values
L	Combination of LIP-LOC and LIP-OPEN	6	P-N, P-W, L-CL, L-CR, L-W, D-CR
T	Combination of TT-LOC, TT-OPEN, TB-LOC, and TB-OPEN	19	D-CR-U-M, A-CL-U-N, A-CL-U-M, A-CR-U-M, A-N-U-M, A-MN-PA-N, A-MN-PA-MN, A-M-PA-M, A-M-U-M, A-W-V-M, P-CR-PA-MN, P-M-U-MN, P-W-V-CL, P-W-V-CR, P-W-V-N, P-W-U-N, P-W-U-MN, P-W-PH-MN, R-N-U-M
G	Combination of VEL and GLOT	3	C-VO, C-VL, O-VO

Table 2.2: Definition of the articulatory feature set used for pronunciation modeling, constructed from the feature set in Table 2.1 by making some simplifying assumptions (see text). For each of the features in this set, the values are combinations of values from the corresponding features of Table 2.1; for example, T = D-CR-U-M corresponds to TT-LOC = D, TT-OPEN = CR, TB-LOC = U, TB-OPEN = M.

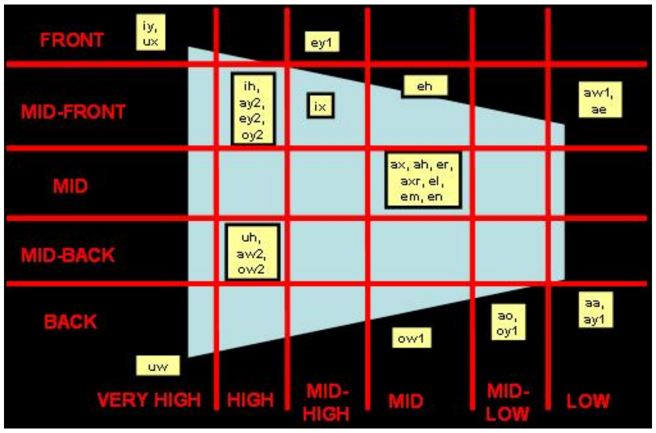


Figure 2.1: Relationship between *vow*, *height*, and *frontness*.

the more forward constriction and *pl2* and *dg2* for the rear constriction. If there is only one constriction, it is always described by *pl1* and *dg1*, no matter how far back in the vocal tract it is. For example, in a /p l/ sequence with a lateral constriction during the burst, we have *pl1 = LAB, dg1 = FRIC, pl2 = LAT, dg2 = CLO*; an /l/ alone would have *pl1 = LAT, dg1 = CLO, pl2 = NONE, dg2 = VOW*. Note that the degree features are not truly physical degrees of constriction; e.g., the same degree of constriction could result in a fricative or not, depending on the pressure behind it. We label a constriction as a fricative only if there is turbulence noise. *Pl2* has one value fewer than *pl1*, because there can not be two labial constrictions.

Height and *frontness* are redundant with *vow*; Figure 2.1 shows the mapping between them. Only *vow* was used in manual transcriptions, but classifiers were trained for all three features. The rationale is that there may be an advantage to training classifiers with a smaller number of output classes, but for the human labelers it was easier to label *vow* directly.

In the glottal state feature, "aspiration" refers to voiceless with aspiration (e.g. the aspirated part of a voiceless stop burst). "Aspiration + voicing" is used for voiced [h] and aspirated vowels/liquids/glides. When we label a segment as "aspirated", we are including aspiration noise that may originate elsewhere other than the glottis (so it is not really a "glottal state", but it is convenient to lump it into this feature).

2.2 Data

This section describes the data sets used in the project. For the most part, we relied on pre-existing sources of data for both audio-only and audio-visual recognition experiments. In addition, prior to the workshop, we collected a new set of manual transcriptions at the articulatory feature level, to be used as reference values for evaluating classifiers and alignments. We briefly describe this new data set here and give further details and analysis in Chapter 5.

2.2.1 SVitchboard 1 data and baselines for acoustic-only experiments

The Small-Vocabulary Switchboard (SVitchboard) database is described in detail in [45]. Here we give a brief overview and present the performance of our baseline recognizers on SVitchboard data.

SVitchboard consists of small-vocabulary tasks, from 10-word to 500-word, defined using subsets of the Switchboard-1 corpus; each task has a closed vocabulary (i.e., an out-of-vocabulary rate of 0%). SVitchboard is intended to facilitate research on new and possibly complex acoustic models for recognition of conversational speech, by enabling first-pass decoding for computationally expensive models. There is no need to use a lattice/N-best rescoring approach, avoiding the complex error analysis arising from the interaction between the errors of the system used to produce lattices, and the novel system used to rescore them. SVitchboard's closed vocabulary further simplifies system building and error analysis. For these reasons, SVitchboard is an attractive choice for articulatory feature-based ASR research, where the new models are computationally demanding but it may still be important to use conversational speech.

Feature	Description	# values	value = meaning

pl1	Place of forward constriction	11	LAB = labial (e.g., [b], [w])
			L-D = labio-dental (e.g., [f])
			DEN = dental (e.g., [th])
			ALV = alveolar (e.g., [d], [s])
			P-A = palato-alveolar (e.g., [sh])
			VEL = velar (e.g., [k])
			GLO = glottal (only for glottal stop)
			RHO = rhotic (e.g., [r])
			LAT = lateral (e.g., [l])
			NONE = no constriction (vowel)
			SIL = silence

dg1	Degree of forward constriction	6	VOW = vowel (no constriction)
			APP = approximant/narrow constriction
			FLAP = flap (e.g., [dx])
			FRIC = fricative (e.g., [s], [th], stop bursts)
			CLO = closure (e.g., [tcl], [m])
			SIL = silence

pl2	*Place of rear constriction, if any*	10	*L-D, DEN, ALV, P-A, VEL, GLO, RHO, LAT, NONE, SIL*
dg2	*Degree of rear constriction, if any*	6	*VOW, APP, FLAP, FRIC, CLO, SIL*
nas	Nasality	2	+, -

glo	Glottal state	6	VOI = voiced (regular pitch periods)
			VL = voiceless
			ASP = aspiration (e.g., [hh], aspirated vowels)
			STOP = glottal stop
			IRR = irregular pitch periods
			A+VO = aspiration + voicing

rd	Lip rounding	2	+, -
vow	Vowel shape	28	aa, ae, ah, ao, aw1, aw2, ax, *axr*, ay1, ay2, eh, *el, em, en*, er, ey1, ey2, ih, *ix*, iy, ow1, ow2, oy1, oy2, uh, uw, *ux*, N/A = not a vowel
ht	Vowel height	7	LOW, MID-L = mid-low, MID, MID-H = mid-high, HIGH, V-HI = very high, N/A = not a vowel
frt	Vowel frontness	6	BK = back, MID-B = mid-back, MID, MID-F = mid-front, FRT = front, N/A = not a vowel

Table 2.3: Definition of the articulatory feature set defined for observation modeling. Italicized values were used in manual transcriptions but not in the AF classifiers. In addition, the AF classifiers have a "silence" output class for each feature.

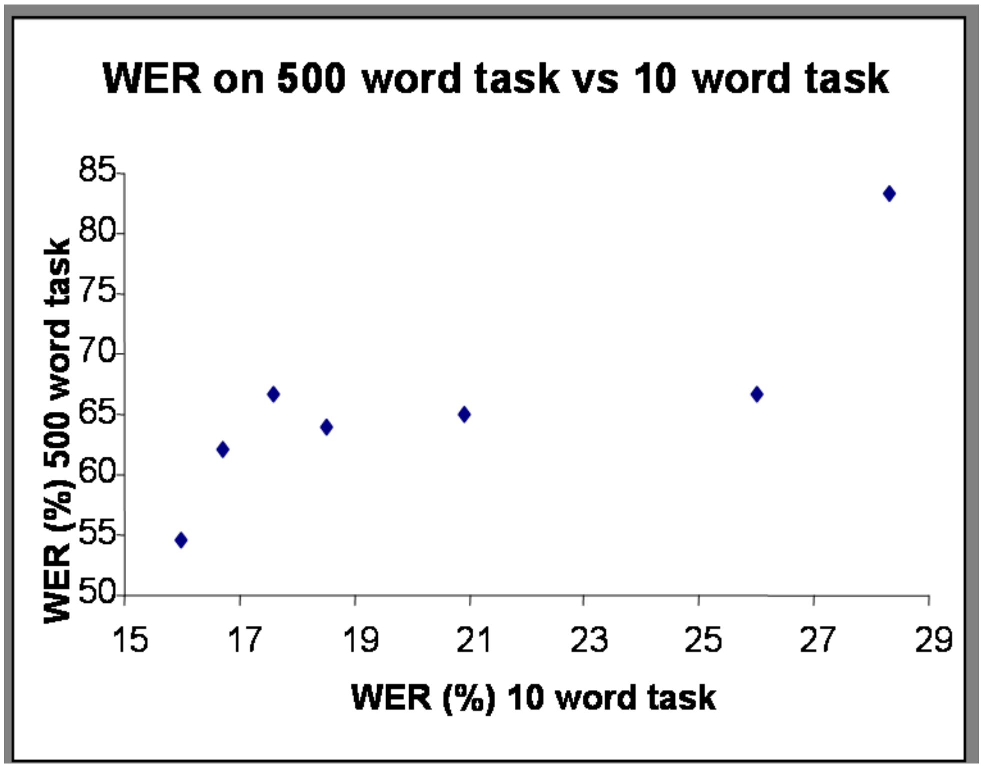


Figure 2.2: 10-word vs. 500-word WERs for a variety of systems built at the workshop.

All of the data in SVitchboard 1 are taken from the Switchboard 1 corpus of two-person telephone conversations [31] and are selected by growing the vocabulary one word at a time, using a simple greedy algorithm described in [45] to attempt to maximize the amount of data available for any given vocabulary size. Other desirable properties in SVitchboard are a minimal number of very low frequency words and at least one token of every word in the training, validation and test sets. Standard partitions of the data are defined for 5-fold cross validation.

For all of our experiments, we used the 10-word and 500-word tasks, and only the first fold defined in the 5-fold cross-validation scheme. This corresponds to using the "A", "B", and "C" sets for training, the "D" set for cross-validation/development, and the "E" set for testing. In most experiments, to save computation time, a smaller development set was used, consisting of 500 utterances (for both the 10- and 500-word tasks) constructed by taking every N^{th} utterance from the full validation set. We refer to the full "D" set as "CV" and the 500-utterance subset as "CV_small". The 10-word task has a total of 3.2 hours of data (0.9 hours of non-silence speech) and the 500-word task has 14.6 hours (6.4 hours of speech).

At the time of SVitchboard's construction, the very small vocabulary sizes (e.g. 10 words) were thought to be useful only for debugging models. However, as shown in Figure 2.2, there is an almost monotonic relationship between word error rates (WERs) on the 10-word task and on the 500-word task, across a variety of model types. Therefore, the best model on the 10-word task is likely to also be the best on the 500-word task. This is good news: Exploratory work can be done on the 10-word task, with some confidence that the relative performance of various systems will be the same when moving up to the 500-word task.

SVitchboard 1 baselines

A set of baselines using whole-word models, built with HTK [42] using Mel-frequency cepstral coefficients (MFCCs), was presented in [45]. The 10-word and 500-word error rates for the train/test division used at the workshop are reproduced from [45] in Table 2.4.

For all of the workshop experiments, the acoustic observations were 13 speaker-normalized perceptual linear prediction coefficients (PLPs), plus derivatives (deltas) and second derivatives (delta-deltas), giving a

39-dimensional observation vector. The observations were computed with HTK. We used subword (monophone or triphone), rather than whole-word, models as baselines for the workshop experiments, as these are more comparable to the AF-based models. We used baseform dictionaries allowing for up to three baseform pronunciations per word. The dictionaries and phone set were based on ones from the Spoken Language Systems group at MIT. The language models were the HTK-generated word bigrams provided with SVitchboard [45]

The procedure used for learning Gaussian mixture parameters, here and in all experiments throughout the workshop, was as follows. We begin with single-Gaussian mixtures, initialized with zero means and unit variances. These are trained via EM until a likelihood convergence criterion is reached, typically until the difference in log likelihood between iterations is below a certain threshold (e.g., 2%). Each Gaussian is then "split" into two Gaussians, whose variances are equal to that of the original Gaussian and whose means are slightly perturbed from that of the original. This new model, with twice as many Gaussians, is trained until convergence and the Gaussians are again split. We repeat this split-and-converge procedure, testing each converged model on a CV set. In later iterations, once the number of Gaussians becomes very large and some of their occupancies very small, we only split Gaussians with occupancies above a certain threshold and also remove Gaussians whose occupancies fall below a threshold. Once the CV WER stops improving, we fix the number of Gaussians and continue training until a tighter convergence threshold, typically a 0.2% log likelihood difference between EM iterations, is reached. See the GMTK documentation for further details on the splitting and vanishing options [30].

Monophone and triphone baseline results obtained using this procedure are given in Tables 2.5 and 2.6, respectively. Table cells marked "-" denote conditions not tested.

For most experiments (all but the tandem modeling experiments of Chapter 4), we used the Mississippi State University word alignments for Switchboard [27] during training, as we found that this gave substantial speedups. This also slightly improved recognition performance (2–3% absolute WER reduction for a monophone system on the 500-word task). Table 2.5 gives the monophone baseline WERs for the 500-word task when training without the alignments, for comparison.

The full set of baseline results on the "E" test set, produced before and during the workshop, are summarized in Table 2.7.

Vocabulary size	WER(%)	
	CV	Test
10	20.2	20.8
25	35.6	34.9
500	69.8	70.8

Table 2.4: SVitchboard whole-word baseline WERs for the full CV and test sets.

Vocabulary size	WER(%)		
	small CV	CV	Test
10	16.7	18.7	19.6
500	62.1	-	65.0
500, no training alignments	65.1	-	67.7

Table 2.5: SVitchboard monophone baseline WERs for the small CV, full CV, and test sets. The cells markes "-" are conditions that were not tested.

System	WER(%)		
	small CV	CV	Test
HTK	-	56.4	61.2
GMTK / gmtkTie	56.1	-	59.1

Table 2.6: SVitchboard triphone baseline WERs for the small CV, full CV, and test sets, on the 500-word task.

Model	WER(%)	
	10 word	500 word
Whole word	20.8	70.8
Monophone	19.6	65.0
HTK triphone	-	61.2
GMTK triphone	-	59.2

Table 2.7: Summary of the SVitchboard baseline WERs for the test set.

2.2.2 CUAVE data for audio-visual experiments

Two databases, CUAVE and AVICAR [51], were tested for audio-visual speech recognition experiments. In the end, our recognition experiments were done using only the CUAVE database [71]. AVICAR is an audio-visual database of isolated digits, telephone numbers, and read sentences recorded in variable natural lighting, in a moving automobile under five acoustic noise conditions (wind and traffic). In our initial experiments with AVICAR, we found that the issues of visual variability were insurmountable during the time of the workshop and, ultimately, of lesser interest than the inherent issues of audio-visual integration. We therefore concentrated on CUAVE, which provides a more controlled audio-visual environment.

CUAVE consists of isolated and connected digit sequences recorded in high-resolution video, with good lighting, in a quiet recording studio. Acoustic noise was added at various signal-to-noise ratios (SNRs). In a digit recognition task, baseline audio-only word error rates are 1.5%, and baseline video-only word error rates are 60% (comparable to the video-only WER reported, e.g., by [17]).

Experiments in this project made use of a portion of CUAVE. The task consists of the words “zero” through “nine,” uttered in ten-digit sequences with silence after each word. Despite the inter-word silences, the task is nevertheless a (simplified) connected-digit task: We did not segment the data into isolated words nor constrain the recognizer to hypothesize inter-word silences. Audio observations included 13 Mel-frequency cepstral coefficients (MFCCs), energy, deltas and delta-deltas, giving a 42-dimensional observation vector. Video observations included the first 39 coefficients from a discrete cosine transform (DCT) of the grayscale pixel values in a rectangle including the lips, upsampled from 30 to 100 frames per second (fps).

All recognizers were trained on clean data (no added noise) and were tested at six different SNRs: ∞ (clean), 12dB, 10dB, 6dB, 4dB, and -4dB. Recognizer training consisted of three stages. First, each recognizer was trained using 60% of the noise-free data, with a Gaussian probability density function (PDF) for each combination of hidden state variables. Second, the number of Gaussians per PDF was increased, the recognizer was re-trained using the same data, and the recognizer was tested on a separate 20% of the noise-free data. This second stage of training was repeated as long as WER continued to improve. Third, the recognizer was tested using development data representing each of the six SNRs (∞, 12, 10, 6, 4, and -4dB), and for each SNR, a video stream weight was selected to minimize WER. The optimum video stream weight was typically 0.0 at SNR=∞, 0.3-0.8 at SNR= -4dB, and 0.1 at all intermediate SNRs. The audio stream weight was one minus the video stream weight. WER results in this report are reported using development set data; tests using independent test data were not completed.

2.2.3 Manual articulatory feature transcriptions

The last data set we describe is a small set of manually transcribed utterances at the AF level, newly collected for the workshop.

A major obstacle for AF-based recognition is the lack of data labeled with AF values. Classifiers are typically trained and tested on phonetic alignments converted to feature values. However, the actual value of a given feature may differ drastically from its canonical value. Therefore, one may miss precisely those phenomena one wishes to model with AF-based approaches. This problem is particularly acute in conversational speech, which is characterized by great variability in pronunciation.

One way to ameliorate this problem in training AF classifiers is to use an embedded training approach [85], a version of which we have implemented and will describe in Chapter 4. Using this approach, an initial set of classifiers can be trained based on phone alignments converted to feature values. Then, using the trained classifiers in conjunction with a model of the AF dynamics, a new set of feature alignments (possibly no longer corresponding to canonical phones) is generated. A new set of classifiers is trained using the re-aligned transcriptions, and the process is iterated until some performance threshold is reached. This approach allows for refinement of the original phonetic transcription, but there is still no measure of the quality of the refined transcriptions or of the classifier performance relative to some "ground truth".

There are two other alternatives to automatic phone alignments: Physical articulatory measurements and narrow phonetic transcriptions. Both of these have certain drawbacks. Physical measurements (e.g. [87]) are typically restricted to scripted speech tasks, and, more importantly, are highly speaker-dependent. Converting such measures to linguistically meaningful features is in itself a research project. Manual transcriptions at a narrow phonetic level, such as the Switchboard Transcription Project (STP) [37], contain much of the needed information. However, they can miss some useful information; in the case of STP, examples are unreleased stops and double articulations.

In this collection effort, 78 utterances from SVitchboard, and another 9 from the phonetically transcribed portion of Switchboard (from the Switchboard Transcription Project [37]), were labeled using the eight articulatory tiers of Table 2.3 (not including the redundant tiers "frontness" and "height"). Two transcribers labeled the utterances in a multi-pass strategy. Developing a labeling strategy for this type of transcription was in itself an interesting and challenging task. In Chapter 5, we describe in detail the transcription effort and inter-transcriber agreement measures, and use the data to analyze the performance of our AF classifiers and alignments produced by our models.

2.3 Infrastructure and tools

The main tool used for implementation, training, and testing of our models is the Graphical Models Toolkit [10] (GMTK). In additional to the basic toolkit, we have developed several additional tools for this project. In order to be able to train and test GMTK recognizers in parallel at multiple sites, we developed site-independent distributed training and decoding scripts, which allow the user to specify the local commands for distributing jobs on his/her site's parallel processing environment. The general approach for parallel EM training is, at each iteration, to compute the occupancy statistics for multiple "chunks" of utterances in parallel (the E step), followed by combining the statistics and updating the model parameters on a single machine (the M step). The next two sections describe two other tools, gmtkTie for generalized state tying in a DBN and GMTKtoWaveSurfer, a visualization/debugging tool for examining GMTK variable values using KTH's WaveSurfer [84].

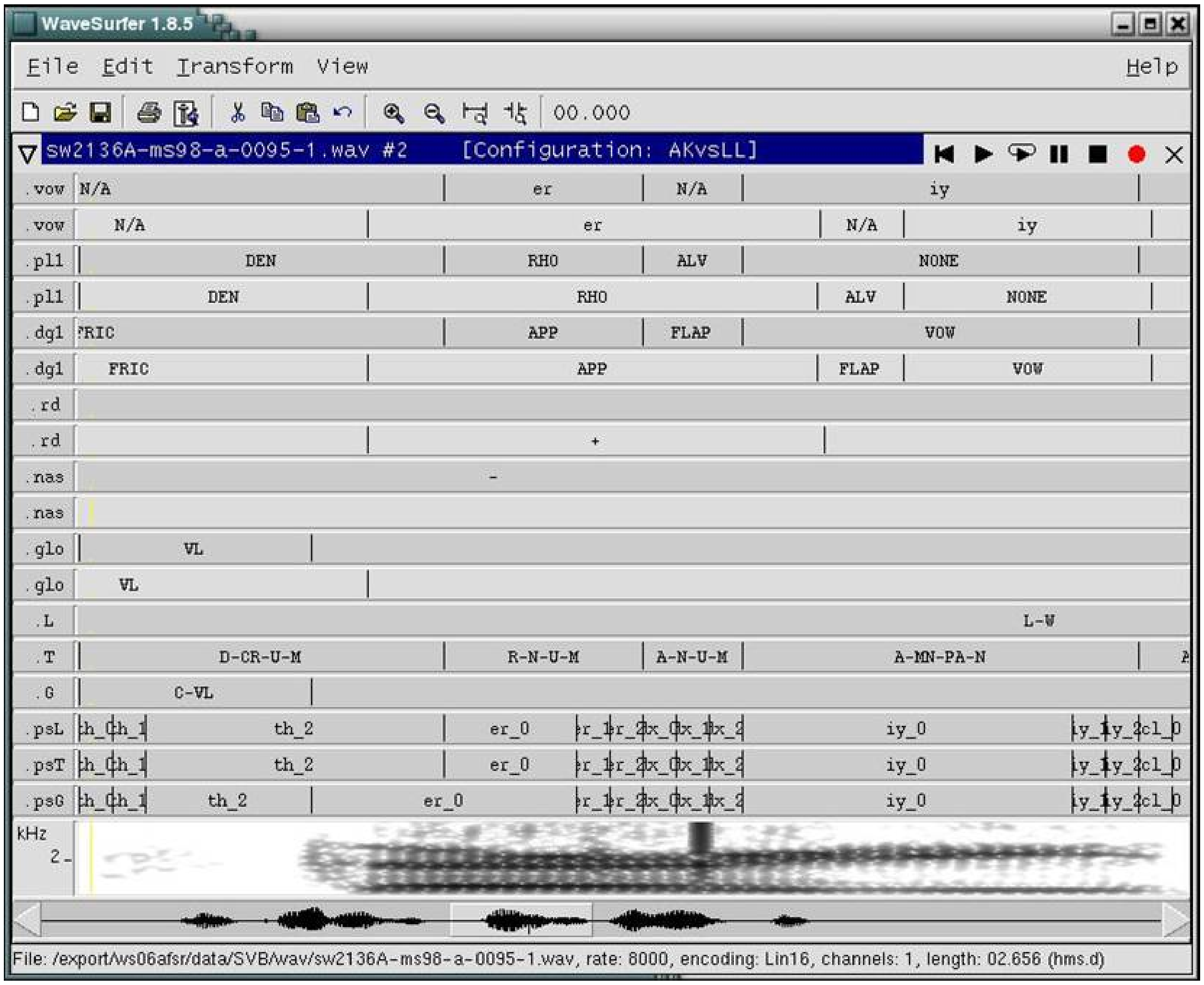
2.3.1 gmtkTie

gmtkTie[2] is part of GMTK and adds parameter tying capabilities similar to those available in HTK. gmtkTie provides manually-controlled, bottom-up, and decision tree-based methods for clustering and then tying parameters. For this workshop, only the decision tree method was used because it can synthesize parameters for unseen contexts. It uses the algorithm developed by Odell [69] that is also used in HTK, although gmtkTie allows more user control over parameters such as the type of distance measure, outlier removal method, and centroid calculation.

gmtkTie is more flexible than HTK's HHEd tool, in that the user can specify the features associated with each parameter. The decision tree method clusters parameters by asking binary (yes/no) questions about the values of these features. In HHEd, the features are limited to left and right phonetic context. gmtkTie can use arbitrary features (so long as the user specifies them) and can also cluster within arbitrary subsets of the model parameters.

We found that a GMTK-based triphone system built using gmtkTie is at least as good as an HTK-based system built using HHEd – the GMTK triphone system for the 500-word task had a WER of 59.2% on the E test set, compared to the HTK system's WER of 61.2%.

[2]https://ssli.ee.washington.edu/ssliwiki/GMTK/gmtkTie


WaveSurfer 1.8.5
File Edit Transform View Help
00.000
sw2136A-ms98-a-0095-1.wav #2 [Configuration: AKvsLL]
.vow N/A er N/A iy
.vow N/A er N/A iy
.pl1 DEN RHO ALV NONE
.pl1 DEN RHO ALV NONE
.dg1 FRIC APP FLAP VOW
.dg1 FRIC APP FLAP VOW
.rd
.rd +
.nas -
.nas
.glo VL
.glo VL
.L L-W
.T D-CR-U-M R-N-U-M A-N-U-M A-MN-PA-N
.G C-VL
.psL th_0 th_1 th_2 er_0 er_1 er_2 dx_0 dx_1 dx_2 iy_0 iy_1 iy_2 cl_0
.psT th_0 th_1 th_2 er_0 er_1 er_2 dx_0 dx_1 dx_2 iy_0 iy_1 iy_2 cl_0
.psG th_0 th_1 th_2 er_0 er_1 er_2 dx_0 dx_1 dx_2 iy_0 iy_1 iy_2 cl_0
kHz
2

File: /export/ws06afsr/data/SVB/wav/sw2136A-ms98-a-0095-1.wav, rate: 8000, encoding: Lin16, channels: 1, length: 02.656 (hms.d)

Figure 2.3: Example WaveSurfer screen shot generated using GMTKtoWaveSurfer, showing the Viterbi decoding hypotheses for some of the variables in an AF-based model. For a subset of the variables (vow, pl1, dg1, rd, nas, glo), reference manual transcriptions are available and are shown in the lighter gray-colored lines.

2.3.2 GMTKtoWaveSurfer: A visualization tool

GMTKtoWaveSurfer is a tool intended to allow a developer to visually examine the values assigned by GMTK to variables in a given model using the publicly available tool WaveSufer [84] from KTH. This serves both as a debugging tool and as a way of examining in detail the model's behavior. For example, one may wish to make detailed utterance-level comparisons between different models, or between model hypotheses and reference transcriptions. In the case of our models, one may wish to examine when the model proposes AF asynchrony, or when different pronunciation variants are chosen.

GMTKtoWaveSurfer mediates between GMTK and WaveSurfer. GMTK can optionally output the values of some or all variable values in a given Viterbi decoding path. However, this output is not easy to visualize. GMTKtoWaveSurfer accepts as input: per-utterances files of GMTK Viterbi variable values; a list of variables of interest; an optional mapping from variable integer values to ASCII labels; and optional reference transcriptions for comparison. GMTKtoWaveSurfer outputs: per-utterance, per-variable WaveSurfer transcription files; and a WaveSurfer configuration file for viewing the transcriptions and (optionally) the references. Figure 2.3 shows an example WaveSurfer screen shot containing transcriptions of variables in one of our AF-based models generated using GMTKtoWaveSurfer.

Chapter 3

Multistream articulatory feature-based pronunciation models

This chapter describes our work on recognition using dynamic Bayesian networks with multiple, possibly asynchronous, hidden streams of articulatory feature states. We have applied these models to both audio-only recognition of SVitchboard utterances and audio-visual recognition of CUAVE digit sequences. In these experiments, we have used the feature set defined in Table 2.2, consisting of the states of the lips (L), tongue (T), and glottis-velum combination (G).

Most of our experiments focused on models with context-independent asynchrony between streams, with synchronization at word boundaries, and without the possibility of features straying from their target values. In addition, we began pilot work on more complex models accounting for cross-word asynchrony, context dependency of the asynchrony probabilities, and substitutions of non-target surface feature values for the underlying target values.

3.1 Model description

Figure 3.1 shows the basic AF-based model used in most of our experiments, based on the pronunciation models of Livescu and Glass [57, 58, 54]. The idea is that each articulatory feature F follows its own trajectory through the state sequence of each word. The sub-word structure for each feature is encoded by variables analogous to those in the phone baseline, and the phone state variable takes on monophone (rather than triphone) values. This means that the observations depend only on the current values of the AFs and not previous or future values. For this reason we refer to this as a "monofeat" model in analogy with the baseline monophone models. Note the similarity between the baseline structure in Figure 1.3 and the feature-specific substructures in Figure 3.1. For clarity, we have again omitted aspects of the structure allowing for multiple pronunciation variants per word.

In addition to this structure, there is an additional variable for each feature (L, T, G) encoding the value of that feature, which is given by a deterministic mapping from phone state to feature values.[1] Since the features now each have their own sub-word state variables, they need not proceed through the word synchronously. The model probabilistically constrains the asynchrony between subsets of the features using the *Async* and *checkAsync* variables. The subsets of features constrained in this way can be chosen arbitrarily. The figure shows the choices used in our experiments, motivated by the intuition that the lips and tongue are more tightly synchronized to each other than the glottis and velum are to the lips and tongue:

- *LTAsync* is a variable whose value corresponds to the current "degree of asynchrony" between the lips and tongue, defined as the absolute difference between $subWordStateL$ and $subWordStateT$. Its distribution encodes the probability of the lips and tongue desynchronizing by a given amount. A maximum degree of allowed asynchrony m is imposed by setting to 0 the probability of $LTAsync = a$

[1]Note that we could equivalently have merged, for each feature F, the *phoneStateF* and F variables.

for any $a > m$. In our experiments, we allowed up to one state of asynchrony between L and T, i.e. $m = 1$ and $LTAsync$ had non-zero probability only for the values 0 and 1.

- $checkLTAsync$ checks that $|subWordStateL - subWordStateT| = LTAsync$.
- $LTGAsync$ is the degree of asynchrony between the mean sub-word state of L and T on the one hand, and the sub-word state of G on the other.
- $checkLTGAsync$ checks that $|\frac{subWordStateL+subWordStateT}{2} - subWordStateG| = LTGAsync$.

We could have used a structure having a single variable for each asynchrony constraint (as in [57]), but its distribution could not be trained via EM. The model as shown can therefore be used for both decoding and training (where the word variable is observed in training).

In this model, we force the AFs to synchronize at the end of each word. This is enforced by the following variables:

- $wordTransitionF$: The word transition variable for each feature F. It is equal to 1 when $stateTransitionF =$ 1 and $subWordStateF$ is the last state in the current word.
- $wordTransition$: Equal to 1 when all of the $wordTransitionF$ variables are 1.
- $checkWordTransitionF$: Checks that $wordTransitionF = wordTransition$. Without this constraint, it would be possible for $wordTransition < F >$ to be 1 for multiple frames before the actual end of the word. We do not know if this would make a significant difference in performance, but it would change the meaning of the word transition variables.

3.2 SVitchboard experiments

In this section we describe the experiments performed using the structure of Figure 3.1. All of the models used are "monofeat" models, as described above, and are therefore compared to monophone baselines. The training procedure starts with single-Gaussian mixture observation models and increases the number of Gaussians incrementally, as described in Section 2.2.1. In addition, we found that different models may vary greatly in the optimal settings of the language model scale and penalty. Along with the number of Gaussian components, we therefore also tuned the language model scale and penalty on the development set separately for each model variant; each model typically required tens of development set decoding runs. As an example, Figure 3.2 shows the development-set WERs of two systems as the language model scale and penalty are varied.

In the first set of experiments, we use a model in which all three states of each phone map to the same feature values, referred to as the "1-state monofeat model". Intuitively, this should be sufficient if the feature state space accounts for all possible articulations and if our model of the dynamics of the AF values is correct. Both of these assumptions are, of course, far from correct, and this model performs much more poorly than the baseline 3-state monophone models. We therefore also explore a 3-state version of the model, analogous to the 3-state monophone model. In the 1-state monofeat, the underlying phone model has three states per phone just as in a monophone model, but all three states map to the same set of articulatory feature values. When no asynchrony is allowed, this is equivalent to a monophone model, except: There is a single state per phone, with a minimum duration of 3 frames per phone; there is a slightly reduced phone set because multiple phones sometimes map to the same AF values; and the transition probabilities are "counted" three times instead of once. In the 3-state monofeat model, each articulator has separate beginning, middle, and end states for each phone. When no asynchrony is allowed, this is equivalent to a monophone model, again with a slightly reduced phone set and with the transition probabilities being counted three times.

Results for the 1-state and 3-state monofeat models are given in Table 3.1. The 3-state monofeat model is a large improvement over the 1-state model, reaching almost the same word error rate as the monophone baseline. However, in both cases the word error rate became worse when asynchrony was allowed. A number of approaches were attempted to improve on this. A noticeable problem in the above experiments was that the state occupancy during EM training was extremely low for the asynchronous states (i.e., those AF

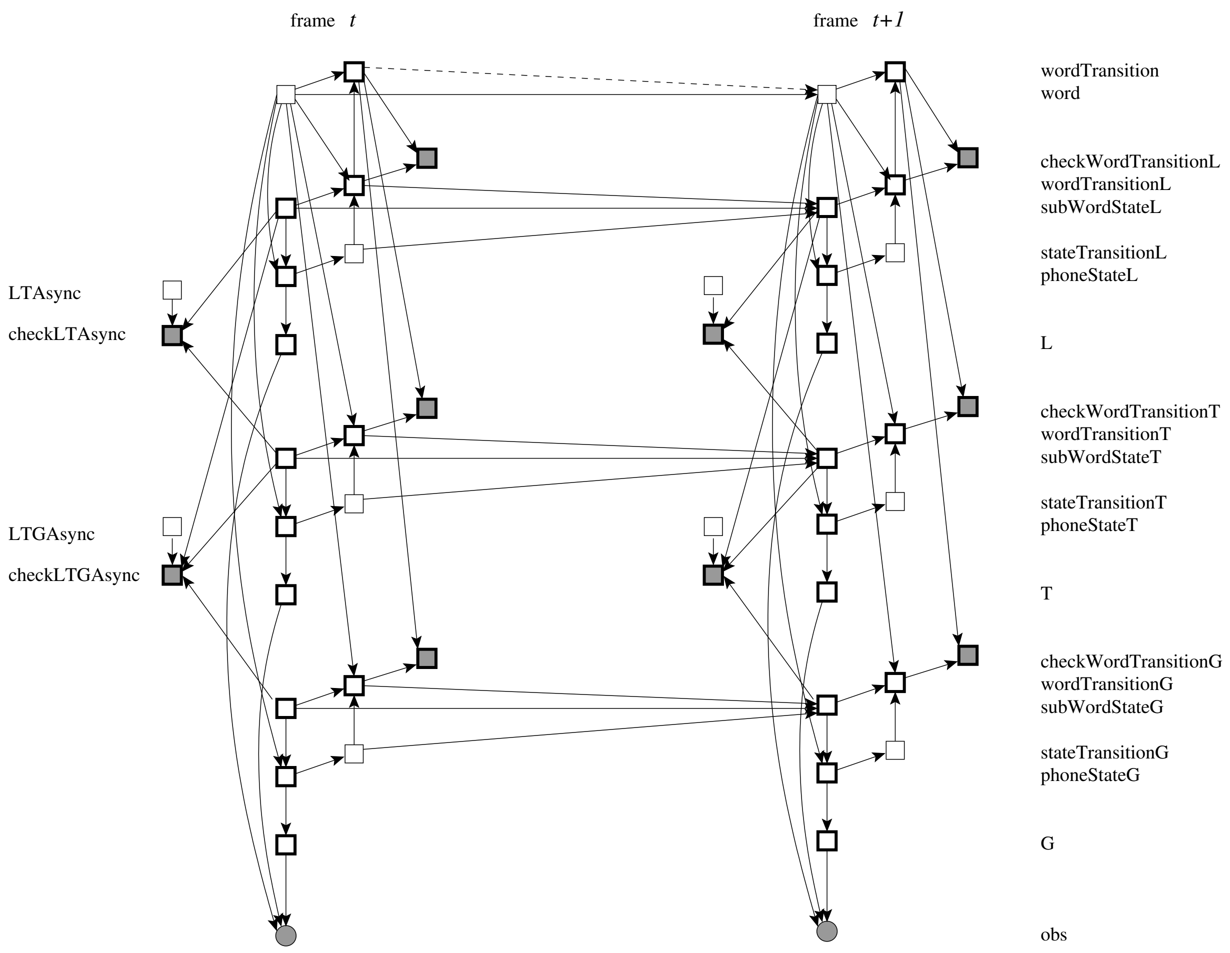


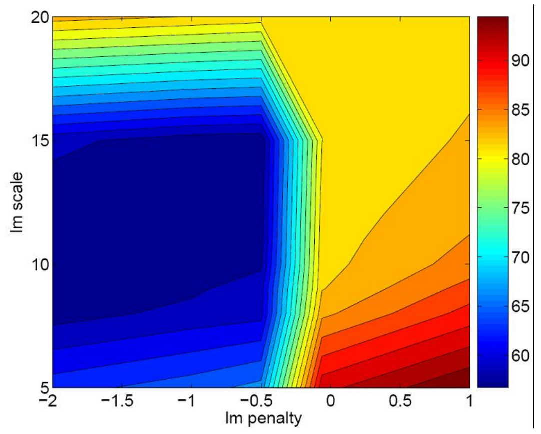
Figure 3.1: An articulatory feature-based DBN for decoding.

combinations that do not correspond to any canonical phone, but are created when the AFs are allowed to de-synchronize). This indicates that either these states are not occurring or we are not modeling them well. This is never a problem in a synchronous model because there are no optional states and the model forces all states to have some occupancy.

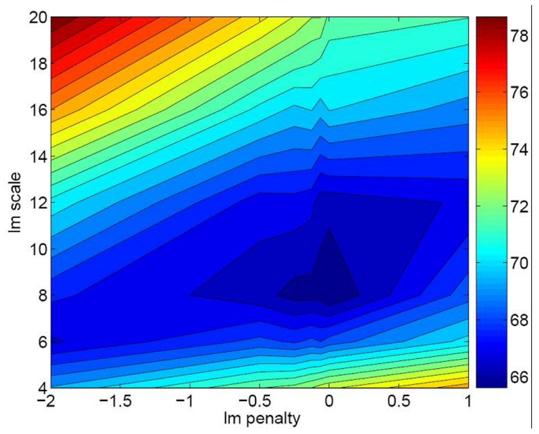
The first attempt to remedy this was to modify the initialization of the parameters. In the monofeat experiments, the asynchrony parameters (i.e., the distributions of the "async" variables) are initialized to uniform values and are learned normally along with the other parameters through the entire training process. In the next set of experiments, the asynchrony parameters were fixed from the initial single-component mixtures through the creation of 8-component mixtures. These parameters were then allowed to train as usual through the rest of the training process. This experiment was done twice; the first time the parameters were fixed at 0.5/0.5 probability of being synchronous/asynchronous, and the second time with probabilities of 0.6/0.4. The motivation for this is that it would force all states to have some significant occupancy, so as to initialize all of the mixtures with reasonable starting values. This experiment successfully gave more occupancy to all states, but failed to improve the error rates for the asynchronous models; see Figure 3.3 and Table 3.2.

The last set of experiments attempted to resolve the occupancy problem through state clustering, using gmtkTie's decision tree-based clustering. The questions are the same as in Section 3.4, and the clustering procedure imposes a minimum occupancy per cluster. However, the word error rate for the asynchronous system did not improve (see Table 3.2).

These are our first attempts and it is possible that they could be improved upon with further exploration of initialization options or decision tree questions. In particular, it is likely that the decision tree questions



(a)



(b)

Figure 3.2: Development set word error rates for two recognizers as the language model scale and penalty are varied.

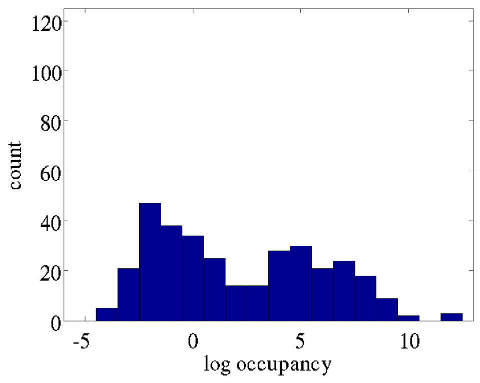
could be improved, as this is the first time that states defined via AF values have been clustered in this way, and the types of questions being asked are quite different than in a typical triphone-based system. Nevertheless, resolving the issue of low-occupancy states remains the most pressing problem for this type of model.

Model	**Vocab**	**CV WER (%)**	**Test WER (%)**
Monophone	10	16.7	19.6
1-state monofeat		28.3	28.5
3-state monofeat, sync		18.5	20.7
3-state monofeat, async		17.6	21.3
Monophone	500	62.1	65.0
1-state monofeat		73.1	74.8
3-state monofeat, sync		64.0	65.2
3-state monofeat, async		66.6	67.4

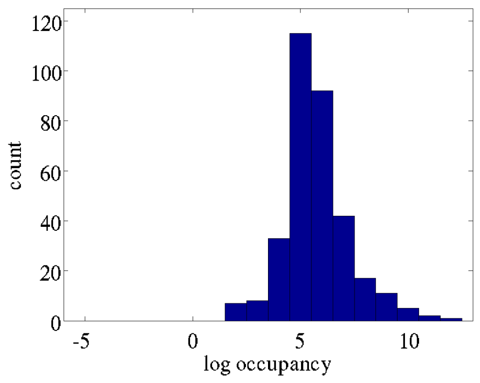
Table 3.1: Test set word error rates of the baseline and "monofeat" models on the "CV_small" set and on the test set.

3.3 Modifications to the basic models

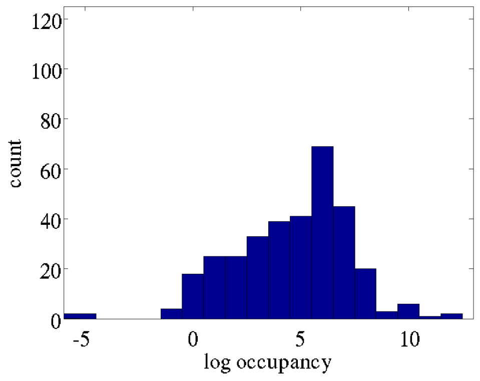
In parallel to experiments with the AF-based models described above, we began experimenting with several modifications: conditioning the asynchrony probabilities on context such as part of speech; relaxing the constraint of complete AF synchronization at word boundaries; and modeling the probabilities of feature substitutions, i.e. of a given AF not realizing its intended value. These modifications resulted in more computationally complex models that did not allow for complete training and testing cycles during the workshop. Below we describe two of these modifications.



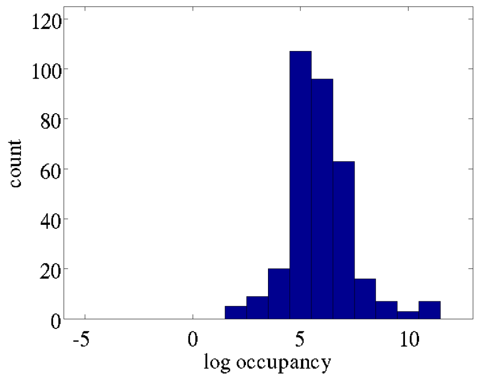
(a) Unaltered initialization, 1 component



(b) 0.6/0.4 initialization, 1 component



(c) Unaltered initialization, final parameters



(d) Unaltered initialization, final parameters

Figure 3.3: Comparison of state occupancies using standard initialization of asynchrony parameters versus the two-stage initialization procedure (for a 10-word system).

Model	CV WER (%)	Test WER (%)
3-state monofeat, async	17.6	21.3
0.5/0.5 initialization	19.5	21.6
0.6/0.4 initialization	19.2	21.9
tied	19.2	22.7

Table 3.2: Test set word error rates for standard training, training with a two-stage initialization, and tied models using decision tree state clustering.

3.3.1 Cross-word asynchrony modeling

One of the difficulties in continuous speech recognition is modeling the effects of cross-word context on pronunciation. In the case of phone-based models, this is usually done using cross-word triphones (or higher-order n-phones), which allow a single phone to have different sets of parameters depending on the neighboring phones. In an articulatory model, many contextual effects can be modeled by simply allowing for asynchrony between the articulatory streams. This has the potential to be especially powerful when modelling cross-word effects in continuous speech, where word boundaries can become unclear. A good example of this is the common pronunciation of "green beans" as "greem beans". Phonetically, the nasal alveolar /n/ is pronounced more like the bilabial [m] when it is followed by a bilabial plosive, such as /b/, and the triphone for /n/ in a /b/ context can capture that. However, this may be inefficient. Although we have training data for [m] from the phoneme /m/ itself, this is not used in training the model for /n/ in a /b/ context since the phones are different. In an articulatory model such as the ones introduced in this chapter, we can allow the lip closure for [b] to be realized early, producing the realization of /n/ as [m].

Implementing cross-word asynchrony in our models while maintaining computational feasibility for GMTK proved to be a nontrivial task. Initial work was done by giving each of the three articulatory feature streams their own word variable. While this model is very general, but also allows for more asynchrony between the streams than is probably necessary; coarticulation effects rarely cross more than one word. Ultimately this model was also very inefficient and abandoned in favor of a more constrained model.

The next approach was to keep the next word as a buffer in each frame. Each articulator then need only keep track of whether it is in the current word or the next word as a binary variable. We used a "two-word state" variable for each AF, indicating where in the two-word state sequence the articulator is. Then the asynchrony constraint variables, LTAsync and LTGAsync, are used to limit how much the two-word state variables can differ across articulators, between the beginning of the current word and the end of the next word.

This model was much more efficient. At the same time, it was more difficult to make use of the training word alignments in the cross-word model than in the original AF-based model. Since the articulators are no longer constrained to be in the same word, it is not clear how to interpret a word "boundary" in the training alignments. The word variable can no longer be considered observed during training. The approach taken was to allow each articulator to make a word transition in a window of plus or minus two frames around the word boundary in the alignment. Even with this speedup, triangulation proved crucial for improving the computational cost of inference on the ten-word task. With the best triangulation found during the workshop, training was about 60 times slower than the original AF-based model, taking up to 160 compute hours for each EM iteration. Future work could investigate alternative cross-word modeling structures.

3.3.2 Substitution modeling

Most of the models we explored did not account for the possibility of features failing to reach their target values. This clearly occurs in spoken language, for example when stops fail to be achieve full closures in words such as *probably*, which may be pronounced much like "prowably" or "prawly". During the workshop we began work on a substitution model that allows for this possibility.

We modeled substitution by adding a variable representing the surface value of each feature as a child of the target, or underlying, value variable (see Figure 3.4(b)). This basic model allows for context-independent substitution, as the surface value of a feature in the current frame is unrelated to its value (or any other variables) in other frames. As may be expected for a context-independent model–allowing each articulator to jump between values arbitrarily often between frames–this model yields poor results. We constructed an alternative model allowing for context dependency.

The context-dependent substitution model is based on the 3-state monofeat model. It differs from the context-independent substitution model in that each surface AF variable depends on the surface AF variable in the previous frame as well as on the current target value (see Figure 3.4(c)).

To simplify the computational complexity of this model, as well as to isolate the influence of substitution modeling itself, we disallowed asynchrony in this model. However, even without asynchrony, computational resources were the main constraint on the model's development. We explored several avenues for overcoming this difficulty, such as constraining the distributions of the surface AF variables to be sparse, exploring various triangulations of the structure, initializing the model with parameters from a similar trained model, and experimenting with beam pruning.

The conditional probability tables (CPTs) of the surface AFs are quite large, especially for the tongue variable (the table has a cardinality of 38 x 57 x 57). Training is computationally impractical if we allow all entries in the CPT to be non-zero. We therefore limited the range of the surface variable such that it could only take values between the previous surface value and the current target value. For instance, if the lips were previously closed, and the current target for the lips is narrow, then the surface value of the lips could be closed, critical or narrow. We also required the surface value to be in the same sub-phone state (beginning, middle, or end) as the target value. We initialized the values in the CPT with a 0.8 probability of the surface value reaching the target value and divided the remaining probability evenly among the other allowed values.

Even with very sparse CPTs, training was too computationally demanding to be feasible. Finding a good triangulation for the training structure turned out to be crucial. With the best triangulation found during the workshop, one utterance could be completed in under three minutes using less than 2G of memory.

In a further attempt to limit computation necessary to obtain a trained model, we initialized the substitution model using parameters from the already trained 3-state monofeat model. For states that did not exist in the 3-state monofeat model, we found the three closest states and averaged their parameter values.

Finally, we used a pruning beam while training. We experimented with various beam widths to find one that was small enough to limit time and memory usage sufficiently and yet not have an impact on the resulting log probabilities. We found that for this model, using GMTK's "-cbeam" and "-sbeam" options with values of 75 for both allowed us to speed up training while not significantly changing the log likelihood values.

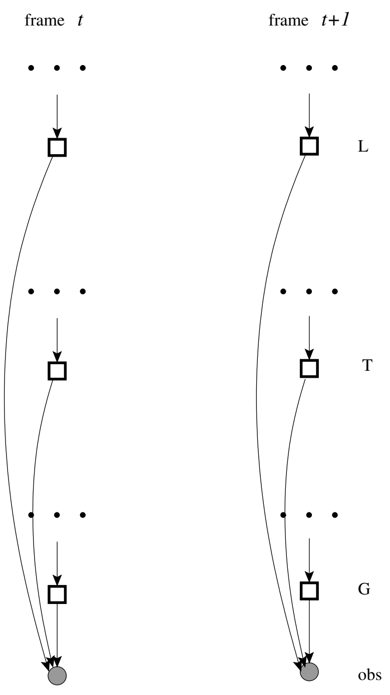
We trained the model up to 128 Gaussians. We did not test the model due to time constraints, but generated forced alignments using the GMTKtoWaveSurfer tool (Section 2.3.2) to explore whether the model is learning the behavior we would like to see. The results are encouraging, though, of course, not conclusive. Figure 3.5 shows a forced alignment of a section of the word "yes". The figure shows the standard feature values and the forced alignments from our model for the lips and tongue. We can see that our model shows the tongue taking extra time to open from the narrow position of the /y/ to the mid position of the /eh/. This is the type of behavior we would like our model to account for. Figure 3.6 shows a segment of the phrase "oh really". In this case, our model marks the lips as remaining protruded through the /r/ of "really". This rounding of the /r/ is realistic in this situation, and again is the sort of behavior we would like to model.

Though we were not able to obtain word error rates from our preliminary work on substitution modeling, qualitative exploration of the forced alignments produced with the context-dependent model indicate that it may be a promising approach. The primary limitations on work in this area are again computational, as the substitution model requires large amounts of both time and memory to train.

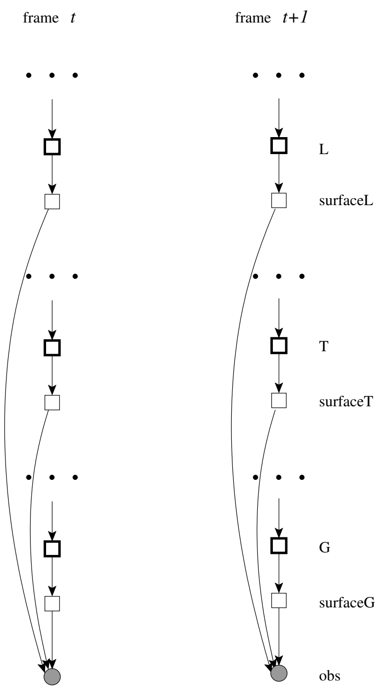
3.4 Audio-visual recognition

In this section, we apply articulatory pronunciation models to the task of audio-visual speech recognition.[2] Articulatory targets corresponding to any given phone may be reached at different times by different articulators. If one articulator is more highly correlated with the video stream, while another articulator is more highly correlated with the audio stream, then inter-articulator asynchrony may cause apparent asynchrony

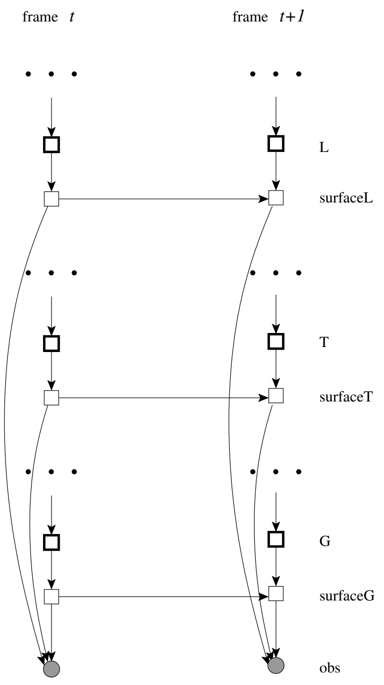
[2]Parts of this section have appeared in [40].



(a) Basic AF-based model.



(b) AF-based model with context-independent substitution.



(c) AF-based model with context-dependent substitution

Figure 3.4: Comparison of AF-based models without substitution, with context-independent substitution, and with context-dependent substitution.

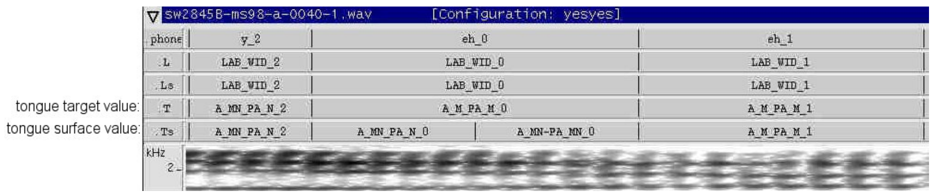


Figure 3.5: A forced alignment of a portion of the word "yes". Our model marks the tongue as taking extra time to transition from the narrow position of the *y* to the mid position of the *eh*.

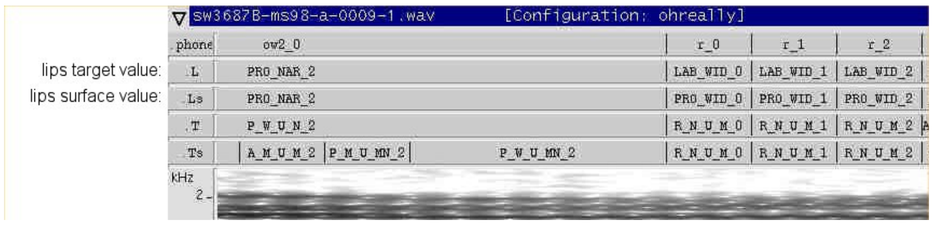


Figure 3.6: A forced alignment of a portion of the phrase *oh really*. Our model marks the lips as remaining protruded through the *r*.

between the audio and video observation streams. Other authors have shown that the word error rate (WER) of an audio-visual speech recognizer may be reduced by allowing asynchrony between the phone states of the audio and video feature streams [17, 36]. Here we demonstrate that the same WER reductions can be achieved by modeling asynchrony among the articulatory states of the lips, tongue, and glottis/velum.

3.4.1 Background

When listening to speech under very noisy listening conditions, humans are able to recognize the phonemes being spoken with much higher accuracy if they can see the lips of the talker [83]. Even under quiet, nearly optimal listening conditions, human subjects who can see the lips of the talker will integrate audio and video information to form the most likely integrated phoneme percept; this has been shown in experiments with anomalous audio and video stimuli, in which subjects cannot correct an incorrect percept even when they know the stimuli are anomalous [60]. Deaf subjects allowed to feel the face of a talker exhibit a comparable integration of visual and haptic cues [25]. Recent research has elaborated some of the neural and cognitive mechanisms that allow human listeners to pre-consciously integrate audio and video speech percepts. Sentence context can change the integrated percept, suggesting that sentence context is incorporated pre-consciously into speech perception [35, 34]. Audio-only speech perception may require less activation of the speech motor areas than audio-visual speech recognition: Watching a talking head in a quiet setting activates only the temporal lobe of a subject in fMRI, but watching a talking head in noisy conditions activates both temporal lobe and pre-motor speech areas [78], as does silent lip-reading [14, 13].

Not all phonemes are visually distinguishable [86, 24]. Typically, consonants with different places of articulation are distinguishable, but consonants produced with the same place of articulation are not (e.g., /t,d,n/ are not distinguishable). A "viseme" is a set of phonemes that are not mutually distinguishable; typical viseme sets for English include about 20 visemes [43].

A number of different techniques have been proposed for integrating audio and video cues in automatic speech recognition. The simplest to implement is feature concatenation: A vector of video features, computed once per 33ms, is upsampled to a 10ms sampling period, then concatenated with the corresponding vector of audio observations [1]. Integration of audio and video modalities in this way suffers from over-confidence of both the audio and video modalities. Integration of audio and video features once per 10ms is termed "fully synchronous" audio-visual speech recognition. Another type of fully synchronous audio-visual speech recognition was proposed by Yehia et al., who used principal components analysis to map a concatenated

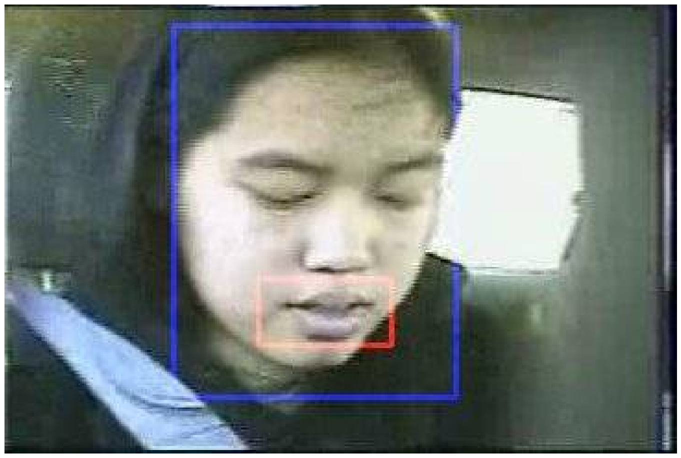


Figure 3.7: An example of audio-visual asynchrony (from AVICAR). The audio signal is silent. The talker has prepared her tongue tip and lips, respectively, for the first and second phonemes of the word "three."

vector of audio and video observations into a smaller vector of pseudo-articulatory parameters [88].

A talker produces a phoneme with just one vocal tract, so the audio and video evidence for each phoneme are, in one sense, perfectly synchronous. Many phonemes, however, do not constrain the positions of all articulators. Figure 3.7, for example, shows a frame excised from the silence preceding the word "three." The talker's tongue tip is already in position for the first phone (/θ/), but the audio signal is still silent, because her lungs have not yet started to push air through the vocal tract. The phoneme /θ/ has no required lip position, so the lips have already rounded in anticipation of the /r/ (the second phoneme of the upcoming word). At this instant, the audio and video modalities may be said to be "asynchronous:" the audio observation is still characteristic of silence, but the video modality exhibits a blend of the phones /θ/ and /r/.

Massaro proposed modeling audio-video asynchrony by running two recognizers in parallel (an audio-only and a video-only HMM), and combining the resulting transcriptions with a fuzzy logic ruleset [59]. Adjoudani and Benoit proposed a dynamic Bayesian network called a "Boltzmann zipper" with partially decoupled audio and video modalities [1]. Neti et al. [66] proposed a "product HMM" (PHMM) in which the number of states per phone model is the product of the number of allowed audio states times the number of allowed video states. Chu and Huang [17] used coupled HMMs (CHMMs) in which the audio and video observations are separately recognized by audio and video HMMs with coupled transition probability matrices.

The PHMM, CHMM, and Boltzmann zipper are all inaccurate models, in the sense that it is not the audio and video streams that are asynchronous; rather, asynchrony exists among the different articulators. In an articulatory phonology representation, Figure 3.7 does not exhibit asynchrony between the audio and video channels, but rather among the articulatory positions of the lips, tongue, and glottis/velum. Figure 3.8 depicts an articulatory phonology-based interpretation of this example.

Niyogi, Petajan and Zhong [67] proposed that audio-visual speech recognition may best be approached using articulatory models. Our research during WS06 implements some of the basic ideas proposed by Niyogi et al. and others. In doing so, we build upon previous systems, including the video-only articulatory feature-based recognizers trained and tested by Saenko and Livescu [76, 77]. In these systems, the "viseme" state variable was factored into loosely synchronized streams representing features such as lip closure, lip rounding, and labio-dental articulation. Factoring the viseme stream improved recognition accuracy in video-only speech recognition tasks.

3.4.2 Models

This section describes a series of baseline and experimental speech recognizers that were tested on the CUAVE database. Audio-visual speech recognition is not nearly as standard as audio-only speech recognition;

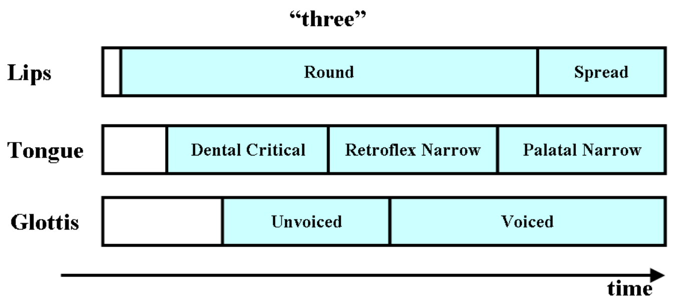


Figure 3.8: An articulatory explanation of Figure 3.7. At one instant during production, all three principal articulators are in different phones of the word "three": The glottis is in phone 0 (silence), the tongue is in phone 1 (/θ/), and the lips are in phone 2 (/r/).

therefore, a relatively large number of "baseline" systems were tested. Section 3.4.2 describes a series of three hidden Markov model-based systems, all of which could have been easily implemented in HTK [42]: an audio-only speech recognizer, a video-only speech recognizer, and an audio-visual speech recognizer with synchronous audio and video streams. The models described in Secs. 3.4.2 and 3.4.2 are both phone-viseme models, but with asynchrony between the two streams.

Hidden Markov models

Three of the baseline systems were standard HMM speech recognizers: audio-only, video-only, and synchronous audio-visual systems. In order to make the implementations directly comparable to the systems that come later, we implemented these as dynamic Bayesian networks in GMTK. Detailed graphs are provided in Figs. 3.9 and 3.10.

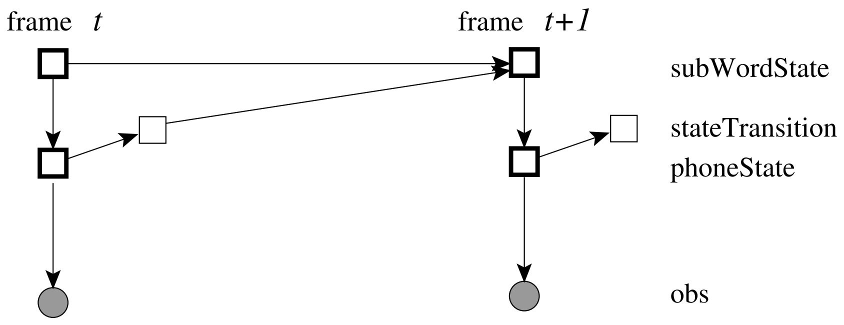


Figure 3.9: Audio-only or video-only HMM-based recognizer.

Figure 3.9 shows a part of the DBN representation for an audio-only or video-only phone HMM-based recognizer. This is identical to the phone baseline DBN of Figure 1.3, but with only a subset of the variables shown to more easily contrast with the various other models we will present. The figure depicts only the variables used to model the relationship between the sub-word state and the observation. The only difference between the two unimodal recognizers is whether the observation vector consists of acoustic or visual observations.

All audio-visual speech recognition experiments used the same uniform (0-gram) language model: Once a word transition occurs, the *word* variable changes to any word in the vocabulary (one through nine, zero, oh, silence) with equal probability. An additional constraint is that we require the recognizer to detect exactly ten non-silence words per utterance.

Figure 3.10 shows an audio-visual speech recognizer with synchronous audio and video observation streams. Although implemented in GMTK, this model could equivalently have been implemented as a two-stream HMM in HTK.

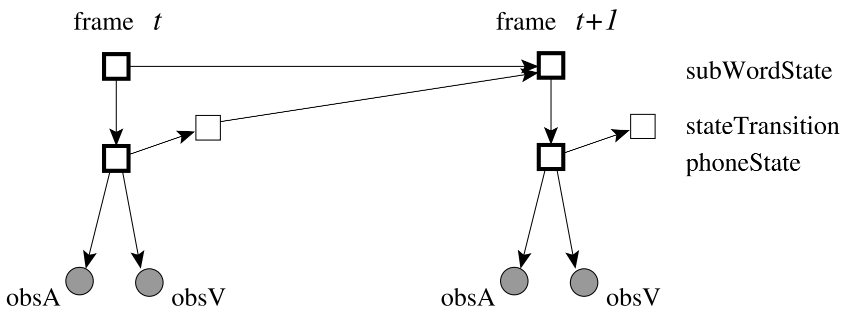


Figure 3.10: DBN representation of audio-visual speech recognition with synchronous audio and video streams. This model is equivalent to a two-stream HMM.

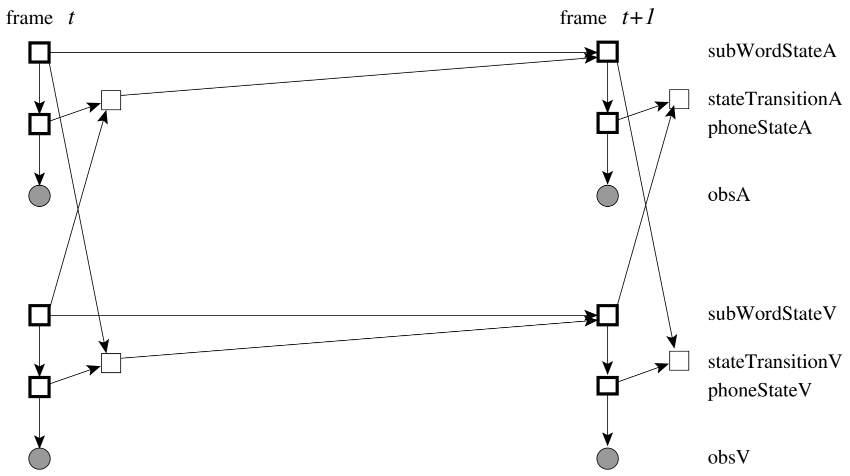


Figure 3.11: DBN representation of audio-visual speech recognition using a coupled HMM. For ease of viewing we have omitted some variables; the model shown correctly depicts the decoding model, but for training we additional need variables to factor the transition probability matrices according to degree of asynchrony of the two streams.

Coupled HMM

Figure 3.11 shows a coupled HMM (CHMM) audio-visual speech recognition model. The language model (not shown) is identical to that of the models in Figs. 3.9 and 3.10. Figure 3.11 has two state variables, $phoneStateA$ for the audio state and $phoneStateV$ for the video state. The two modalities have independent observations, but dependent transition probabilities: The variable $stateTransitionA$ is dependent on the current values of both $subWordStateA$ and $subWordStateV$. In practice, we factor the transition probability table (by means of another hidden variable, not shown), so that the probability of a state transition ($p(stateTransitionA = 1)$) depends only on the current phoneme ($phoneStateA$) and the degree of asynchrony ($subWordStateA$-$subWordStateV$).

To our knowledge, the implementation schematized in Figure 3.11 is the first implementation of a phone-based CHMM with asynchrony that crosses phone boundaries. The experiments of Chu and Huang [17, 18] used whole-word CHMMs. Neti et al. [66] implemented phone-based product HMMs, which are similar, but without cross-phone asynchrony. The model in Figure 3.11 can represent inter-chain asynchrony that crosses phoneme boundaries, but not asynchrony across a word boundary.

Constraining asynchrony with explicit asynchrony variables

Figure 3.12 shows a model of audio-visual asynchrony that we will call a “constrained hidden asynchrony model” (CHAM). It has been previously applied to AVSR by Saenko and Livescu [75]. In a CHAM, asyn-

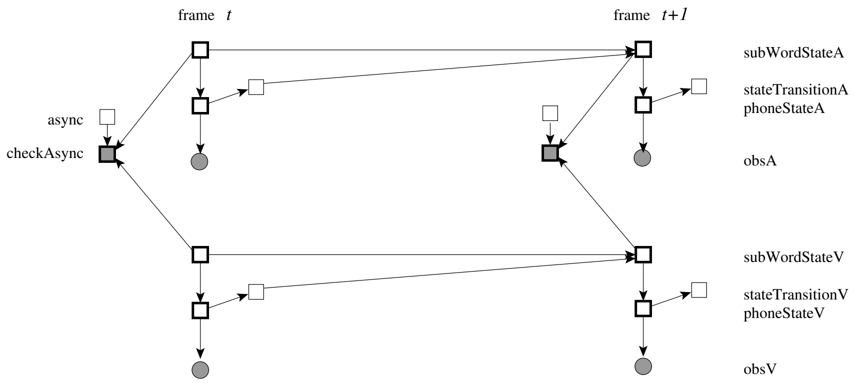


Figure 3.12: DBN representation of audio-visual speech recognition using an explicit asynchrony check variable.

chrony is represented by means of an explicit *async* variable, as in the audio-only models of the previous sections. The model space of a CHAM is nearly identical to that of the corresponding CHMM, but the CHAM is slightly more parsimonious. As in most machine learning applications, the more parsimonious model (the CHAM, in this case) is preferable if adequate.

Articulatory feature models

Figure 3.13 shows an articulatory feature-based CHAM. The AF-CHAM is similar to the original CHAM (Figure 3.12), with the following exceptions. First, there are three hidden state variables, rather than just two: the three hidden chains are intended to represent lips (*phoneStateL*), tongue (*phoneStateT*), and glottis/velum (*phoneStateG*). Second, both of the observations (audio and video) depend simultaneously on all three hidden state streams. Third, and most important, the cardinality of the hidden state variables is considerably reduced. The cardinality of the *phoneStateA* variable in Figure 3.12 is equal to the number of distinct phone states (three times the number of distinct phonemes, thus $3 \times 42 = 126$), but the cardinality of variable *phoneStateL* is only $3 \times 5 = 15$: *phoneStateL* can take values corresponding to the beginning, middle, and ending of the five types of lip gestures (CLOsure, CRItical, NARrow, WIDe, or SILent). Cardinalities of the variables *phoneStateT* and *phoneStateG* are similarly much smaller than the cardinalities of `phoneStateA`.

Figure 3.14 shows the structure of an articulatory feature-based CHMM. The model in Figure 3.14 is identical to the one in Figure 3.11, except for three differences. First, there are three state chains instead of two. Second, both observations (audio and video) depend simultaneously on all three hidden state chains.[3]. Third, the cardinality of all three state chains is significantly reduced, as discussed above for the AF-CHAM model.

3.4.3 Experimental variations

In addition to training and testing all of the models described in Section 3.4.2, experiments performed at WS06 also tested a number of variations. Some of the tested variations are discussed here.

Context tying

A number of different phoneme sets were tested. The “monofeat” systems, depicted in Figs. 3.13 and 3.14, are comparable to monophone HMM recognizers; even so, they have a large number of possible settings, and many settings were not adequately trained by the available training data. In order to ameliorate the resulting data sparsity, we experimented with state tying using the new tool gmtkTie. Figure 3.15 shows, as an example, the decision tree selected by gmtkTie for the purpose of tying the *obsA* densities in an articulatory

[3]We have experimented with models in which the video observation is independent of the “glottis” state chain. Performance was identical to that of the model shown in Figure 3.14.

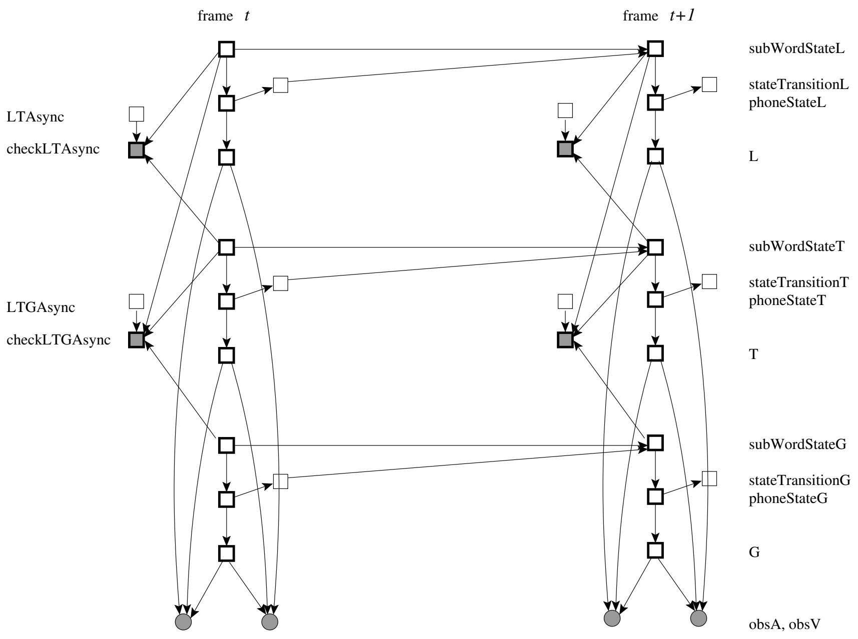


Figure 3.13: An articulatory feature-based audio-visual speech recognizer. Asynchrony variables compute the probability of any given degree of asynchrony between the lips, tongue, and glottis.

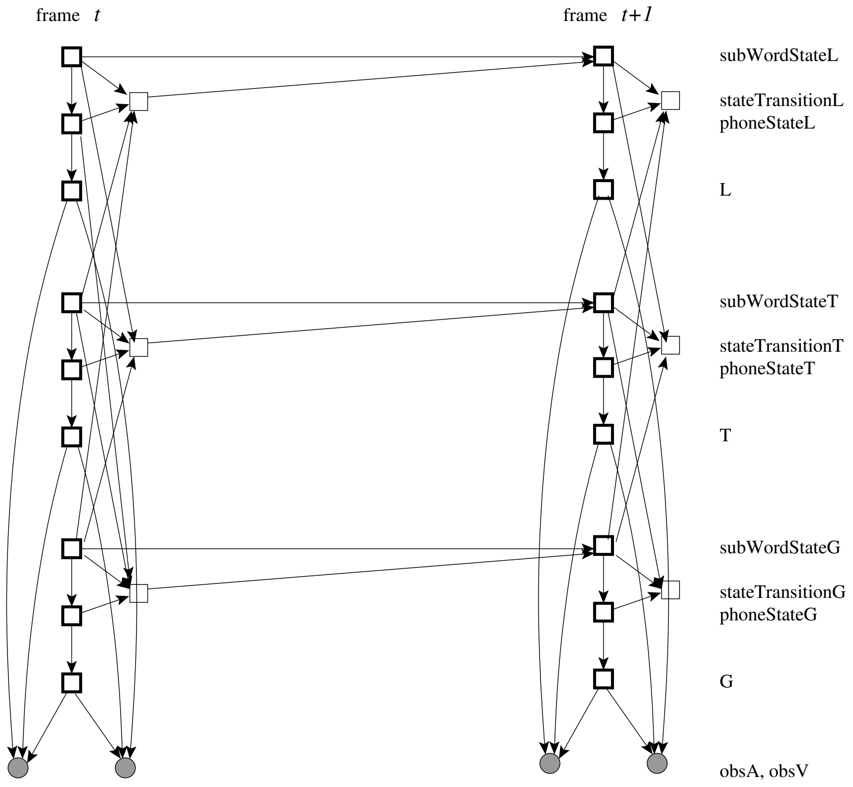


Figure 3.14: Articulatory feature-based coupled HMM. This model differs from Figure 3.11 in that all state variables (Lips, Tongue, Glottis) jointly determine all observation densities (obsA, obsV)
.

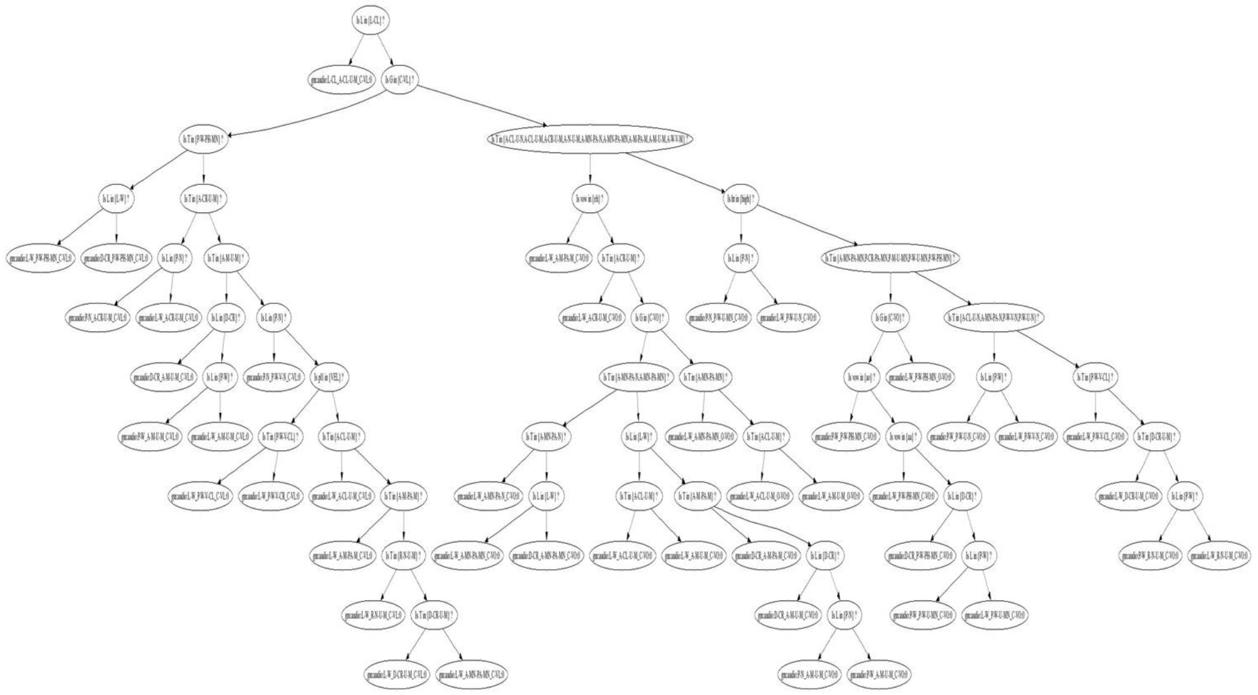


Figure 3.15: Tying tee for the audio PDF.

feature-based CHAM. Figure 3.16 shows the top of the tree, enlarged so that individual questions can be more easily read.

Cross-word asynchrony

None of the models shown in Section 3.4.2 allow the phone state variables (*phoneStateL*, *phoneStateT*, and *phoneStateG*) to be asynchronous across a word boundary. In many cases, the articulators are demonstrably asynchronous across word boundaries (see Figure 3.7, for example, in which the lips and tongue are preparing for the upcoming word "three.") Nevertheless, explicit models of cross-word asynchrony produced higher WER on our CUAVE task than models with only within-word asynchrony. We may speculate that models without cross-word asynchrony are adequate for this task, but for more complex tasks it may be necessary to allow it.

Feature set definitions

For theoretical reasons, most of the experiments performed at WS06 used articulatory feature sets that failed to completely distinguish all of the phonemes used in the baseline system. In one sense, therefore, the baseline system is able to represent phonemes with more precision than the articulatory feature systems. Because the audio-visual task is more constrained than some of the other tasks considered during WS06, we were able to easily test a recognizer with an articulatory feature set expanded in order to uniquely discriminate every phoneme in the baseline system. The modifications to the feature set are listed in Table 3.4.3. The expanded feature set outperformed the base feature set (Table 3.4); therefore, it was used in the systems whose results are reported here (Figs. 3.19 and 3.20).

3.4.4 Recognition results

A large number of audio-visual experiments were performed using the "S" train/validate/test partition of CUAVE. WER results using the validation set are reported in Table 3.4. WER results for some of these

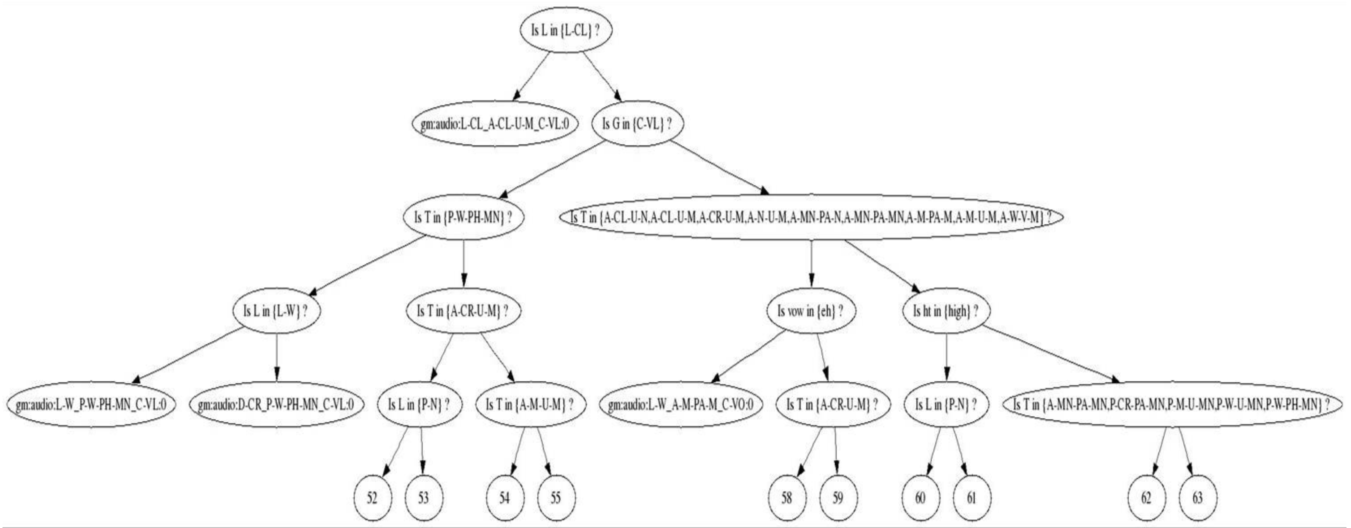

Is L in {L-CL} ?
gm:audio:L-CL_A-CL-U-M_C-VL:0
Is G in {C-VL} ?
Is T in {P-W-PH-MN} ?
Is T in {A-CL-U-N,A-CL-U-M,A-CR-U-M,A-N-U-M,A-MN-PA-N,A-MN-PA-MN,A-M-PA-M,A-M-U-M,A-W-V-M} ?
Is L in {L-W} ?
Is T in {A-CR-U-M} ?
Is vow in {eh} ?
Is ht in {high} ?
gm:audio:L-W_P-W-PH-MN_C-VL:0
gm:audio:D-CR_P-W-PH-MN_C-VL:0
Is L in {P-N} ?
Is T in {A-M-U-M} ?
gm:audio:L-W_A-M-PA-M_C-VO:0
Is T in {A-CR-U-M} ?
Is L in {P-N} ?
Is T in {A-MN-PA-MN,P-CR-PA-MN,P-M-U-MN,P-W-U-MN,P-W-PH-MN} ?
52
53
54
55
58
59
60
61
62
63

Figure 3.16: Tying tree for the audio observation density: enlarged view of the top of the tree.

variable	original setting	expanded settings
phoneStateG	Voiceless	Aspirated, Voiceless
phoneStateT	MidFront	MidFrontTense, MidFrontLax
phoneStateT	HighFront	HighFrontTense, HighFrontLax

Table 3.3: Articulatory feature set modifications necessary in order to uniquely represent all of the baseline recognizer's phonemes in the isolated digits vocabulary.

	SNR					
Recognizer	CLEAN	12dB	10dB	6dB	4dB	-4dB
HMM Systems						
Audio-only	1.3	18.0	23.3	39.7	50.0	81.3
Video-only	63.3	63.3	63.3	63.3	63.3	63.3
Synchronous	1.7	8.0	11.3	23.0	30.0	57.3
Phoneme-Viseme CHAM Systems						
Max Async=1	0.7	8.3	12.7	25.3	35.0	57.3
Max Async=2	1.0	8.7	12.0	22.3	29.3	54.0
Phoneme-Viseme CHMM Systems						
Max Async=1	1.3	6.0	11.3	26.3	31.7	59.0
Max Async=2	1.3	7.7	10.0	20.3	28.3	60.7
Unlimited Async	1.7	7.0	9.0	21.0	31.0	61.3
Articulatory-Feature CHAM Systems						
States/Phone=1	5.0	20.7	24.7	42.7	48.7	68.7
States/Phone=3	2.3	8.0	11.7	23.3	32.7	62.7
Above; Tied	1.3	9.7	12.3	25.3	33.3	65.3
Articulatory-Feature CHMM Systems						
States/Phone=1	8.3	36.0	47.3	-	-	-
States/Phone=3, Max Async=1	1.0	10.3	15.3	27.7	38.0	65.0
States/Phone=3, Max Async=2	1.7	9.3	13.7	28.3	35.7	65.7
As Above; Expanded Feature Set	2.0	6.3	10.3	21.3	29.3	67.0

Table 3.4: Connected digit WER (using the best video stream weight) of several audio-visual speech recognition systems. All systems shown here were tested using the "S" train/validation/test division of the CUAVE corpus.

systems were computed using the "S" evaluation test set; a sampling of those results is given in Figs. 3.17 through 3.20.

Table 3.4 lists WERs achieved by all of the recognition architectures described in Section 3.4.2. Each system was trained using clean audio data. The number of Gaussian mixtures per PDF was increased until WER of the recognizer, computed using clean validation data, reached a minimum. Then, using models trained on clean data, a variety of different video stream weights were tested using validation data from each SNR, and the best video stream weight was selected. The optimum video stream weight was typically around 0.1, except in clean speech (when the optimum video stream weight was 0.0) or at the worst SNR (when the optimum video stream weight varied from 0.3 to 0.8, depending on recognition architecture); the audio stream weight was always set to one minus the video stream weight. The best validation-set WERs are shown in Table 3.4, and the corresponding optimum video weight is shown in parentheses in each column.

The first three lines of Table 3.4 show the WER achieved by HMM recognizers with (respectively) audio-only, video-only, and synchronous audio-visual observations. The next five lines show the results of CHAM and CHMM phoneme-viseme (two-stream) recognizers with varying degrees of asynchrony allowed between the audio and video streams.

The last seven lines of Table 3.4 show WER results, on the validation dataset, using articulatory feature-based recognizers (CHAM-based and CHMM-based). Experiments with articulatory feature systems started with only one state per phoneme, but, as shown in Table 3.4, one state per phoneme was clearly insufficient; all further experiments were performed with 3 states per phoneme.

The last line in the CHAM articulatory system results shows WERs achieved using gmtkTie to combine sparse-data feature combinations into clusters. This result corresponds to the tying trees shown in Figs. 3.15 and 3.16. Although the trees clearly show reasonable questions being asked at each level, the WER results with tying are not better than the results without tying.

Almost all results in Table 3.4 are computed using the same articulatory feature set as was used in

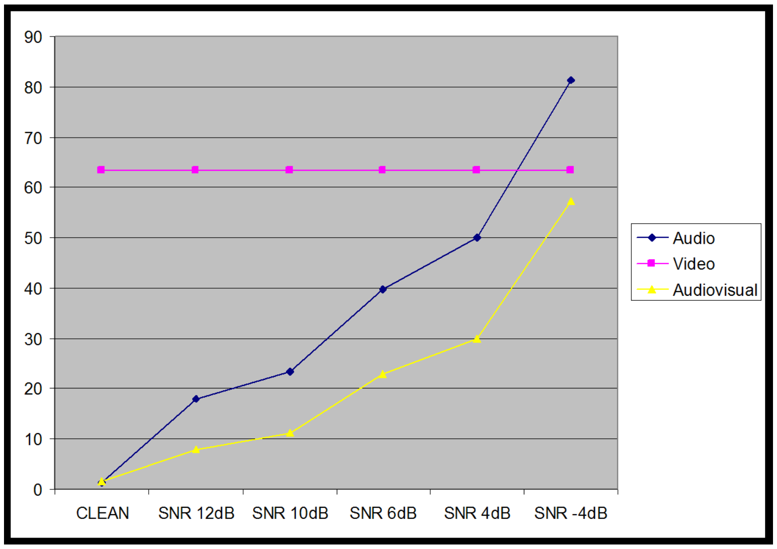


Figure 3.17: WER of video-only, audio-only, and audio-visual HMM recognizers in six noise conditions, "S" train/validate/test division.

SVitchboard experiments. The last line of Table 3.4 was computed using the expanded feature set described in Table 3.4.3. Since the expanded feature set outperformed the original feature set, the recognizer with expanded feature set was used in the results shown in Figs. 3.19 and 3.20.

WER results using evaluation test data have been computed for some of the recognizers shown in Table 3.4; see Figs. 3.17 through 3.20. Figure 3.17 plots the WER of the first three systems in Table 3.4 on evaluation test data, as a function of test-set SNR. The WER of the video-only recognizer is about 60%, independent of acoustic SNR. The WER of the audio-visual recognizer is lower than that of the audio-only recognizer under every condition with acoustic noise.

The next five lines of table 3.4 show the WER of two-stream CHAM and CHMM audio-visual speech recognizers (Section 3.4.2 and 3.4.2). By allowing asynchrony between the audio and video streams, these recognizers achieve lower WERs than the fully synchronous model. Figure 3.18 plots WER as a function of test-set SNR, using evaluation test data, for four different recognizers: a fully-synchronous recognizer (0 states of allowed asynchrony between the audio and video modalities), two coupled HMMs (with maximum allowed asynchronies of 1 state and 2 states, respectively), and an "uncoupled HMM" (a degenerate CHMM in which the degree of asynchrony has no effect on the transition probability, i.e. the two chains are allowed to have an unlimited amount of asynchrony). The best performance is achieved, at most SNRs, by the system with a maximum of 2 states of asynchrony.

The last six lines of Table 3.4 list WER results for articulatory feature-based audio-visual speech recognizers (both CHAM and CHMM systems). Figure 3.19 plots WER as a function of SNR for two models: a phoneme-viseme CHMM with two states of asynchrony allowed between audio and video modalities, and a hidden-articulator recognizer with a maximum of two states of asynchrony allowed between any pair of articulators. The results are almost identical; the differences are not statistically significant.

Despite the similarity in the WERs they produce, the hidden-articulator recognizer and the phoneme-viseme recognizer are not identical. This suggests that combining the system outputs may produce improved performance, and we have tried two system combinations using NIST's ROVER [23]. Figure 3.20 shows the WERs of three recognizers, and of two ROVER system combinations, averaged across all test-set SNRs. The three original recognizers are two phoneme-viseme systems (with maximum allowed asynchrony between chains of one and two states, respectively), and one articulatory-feature system. The leftmost bar shows the results of ROVER system combination using these three systems. The second bar shows the results of ROVER system combination using only phoneme-viseme systems: the articulatory-feature system was replaced by a phoneme-viseme system with unlimited inter-chain asynchrony. ROVER system combination was performed using majority voting: Word boundary times were assigned to each test utterance by the

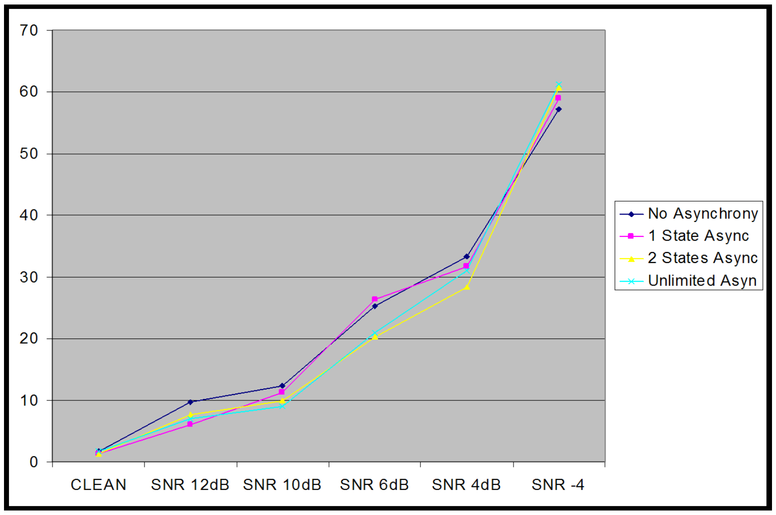


Figure 3.18: WER of phone-viseme CHMM recognizers with maximum allowed asynchrony of 0, 1, 2, or an unlimited number of states, using the "S" train/validate/test division.

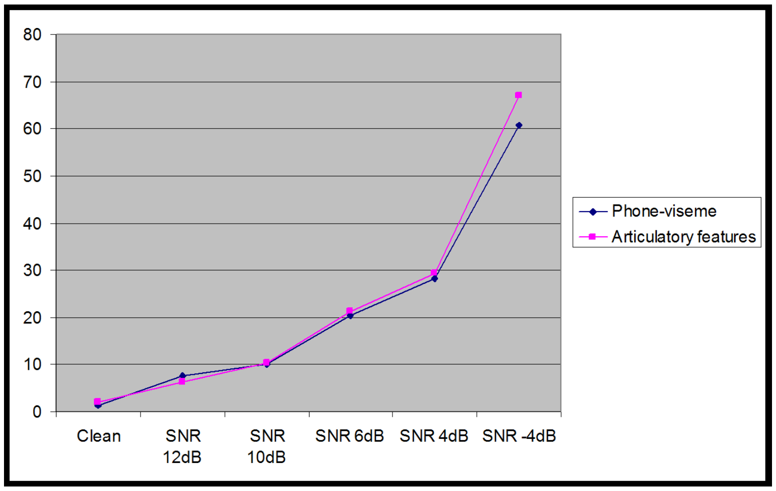


Figure 3.19: WER of phone-viseme and articulatory CHMM recognizers. "S" train/validate/test division.

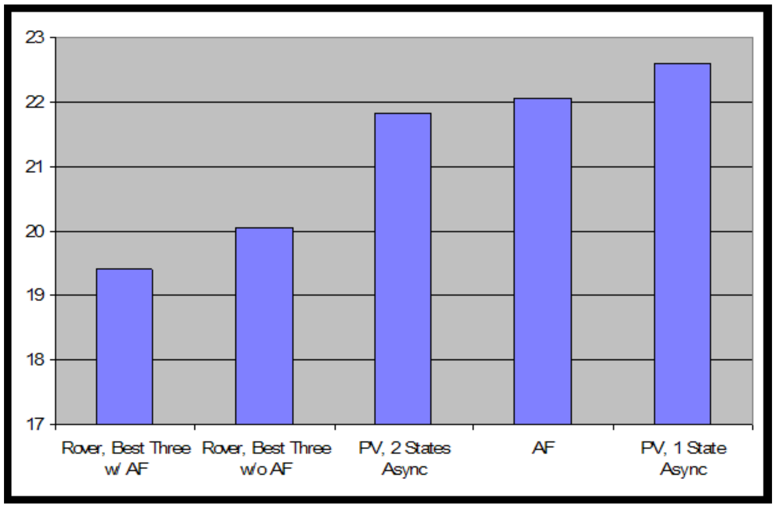


Figure 3.20: WER of three individual systems (two phone-viseme systems and one articulatory feature system) and two ROVER system combinations, averaged across all test-set SNRs. The leftmost bar is ROVER including one AF system and two phone-viseme systems; the second bar is ROVER including three different phone-viseme systems. "S" train/validate/test sets.

maximum-posterior-probability recognizer, and the label of each word was chosen to be whichever word was output for that segment by at least two of the three recognizers. ROVER combination including the articulatory feature HMM produces lower WERs, on average, than ROVER system combination without the articulatory feature system. This difference, however, is not statistically significant on this test set.

Chapter 4

Articulatory feature classifier-based observation models

4.1 Introduction

In previous work by ourselves and others (e.g., [46, 49, 16, 4, 5, 6]), it has been demonstrated that articulatory features can be classified from speech with high accuracy, at least as measured against phone transcriptions converted to AF values. The articulatory feature set typically used in such experiments is similar to our IPA-inspired feature set (see Section 2.1.2. The most commonly used classifiers for this purpose are artificial neural networks (ANNs), or more specifically multi-layer perceptrons (MLPs). The use of a classifier such as an MLP forms the first stage in *hybrid* and *tandem* approaches to ASR, described in the following sections.

Typically, the classifiers are trained independently of the other components of the system. Each MLP (one each for manner, place, etc.) is trained to classify speech into AF values on a frame-by-frame (not segment-by-segment) basis: Given a window of input frames of parameterized speech, the target for each MLP is a 1-of-N encoding of the corresponding AF value. To obtain these targets for the training data, forced phone alignments are computed and mapped to AF values. Although this fails to capture any effects of asynchrony or non-canonical feature values, it is the only practical approach currently available. The phonetic alignments we used were obtained from SRI's Decipher system.[1] The drawbacks of deriving AF labels from forced phone alignments can be mitigated using embedded training; a preliminary successful demonstration of this is given in Section 4.3.3.

The following sections describe the MLP classifiers, hybrid models, and tandem models we experimented with at WS06.

4.2 MLP articulatory feature classifiers

Two sets of MLP AF classifiers were trained by Joe Frankel (U. Edinburgh/ICSI) and Mathew Magimai-Doss (ICSI), one on a large (1776-hour) training set and one on only the SVitchboard training set; the former have been made public and are described in further detail in [26]. The AF set is given in Table 4.1 (repeated for convenience from Table 2.3). A separate gender-independent MLP was trained for each feature. The MLPs are standard three-layer feedforward networks, classifying each frame into one of the values of the corresponding feature. The inputs to each MLP are the PLPs from the current frame as well as those from the previous four and following four frames, giving a 351-dimensional input vector.

The first set of MLPs was trained using a total of 1776 hours of data from Fisher and Switchboard2, excluding Switchboard1 (and thus excluding SVitchboard). We refer to these as the "Fisher MLPs". The number of hidden units for each MLP were selected so as to have a roughly 1000:1 ratio of the number of

[1]Although the alignment system may have included some of our test set in its training set, the effect of this is assumed negligible because of the very large amount of other training data used by Decipher, and because Decipher is only being used to provide phone alignments from which to train AF classifiers.

Feature	**Values**	**MLP accuracy (%)**
place	labial, labio-dental, dental, alveolar, palato-alveolar, velar, glottal, rhotic, lateral, none, silence	72.6
degree/manner	vowel, approximant, flap, fricative, closure, silence	73.6
nasality	+, -, silence	92.7
glottal state	voiced, voiceless, aspirated, silence	85.3
rounding	+, -, silence	84.7
vowel	aa, ae, ah, ao, aw1, aw2, ax, ay1, ay2, eh, er, ey1, ey2, ih, iy, ow1, ow2, oy1, oy2, uh, uw, not-a-vowel, silence	65.6
height	very high, high, mid-high, mid, mid-low, low, not-a-vowel, silence	68.0
frontness	back, mid-back, mid, mid-front, front, not-a-vowel, silence	69.2

Table 4.1: The articulatory feature set and MLP accuracies on the SVitchboard test set.

training frames to the number of parameters. The targets for MLP training are obtained from a deterministic phone-to-feature mapping of forced phonetic alignments from SRI's Decipher system. Training took approximately ten days per MLP, on a 2.2GHz Sun v40z machine using highly optimized software (e.g., multi-threading). The frame-level cross-validation accuracies of the trained MLPs ranged from about 70% to about 90% (see [26] for more details). Table 4.1 gives the accuracies of the MLPs on the SVitchboard test set. All accuracies here are measured against forced phonetic alignments converted to AF values; in Chapter 5 we test MLPs against human labels.

The second set of MLPs were trained on the SVitchboard training data, for comparing AF- and phone-based tandem approaches: (1) a set of AF MLPs identical to the Fisher-trained MLPs above except that the numbers of hidden units were scaled down according to the amount of data, (2) a phone MLP with 46 outputs corresponding to the SRI phone set.

4.3 Hybrid models

So-called "hybrid" HMM-ANN models [62] use a classifier (generally an ANN) to compute the observation emission likelihood for a state, rather than the usual Gaussian mixture model (GMM) used in the standard HMM-based ASR systems. The classifier is trained as described in Section 4.2 and the ANN output activations can be used (after normalization to sum to 1) as an estimate of the posterior probability distribution over classes (generally phonemes in previous work by others). Dividing this by the prior gives a scaled likelihood which can be used directly in place of the state observation emission likelihood.

Using an ANN in place of a GMM has several advantages. Perhaps the two key ones are that the ANN is trained discriminatively and that it can use a long window of input frames. Although ANN training is more difficult than generative HMM training (computationally, and in terms of heuristically setting parameters such as learning rate), once trained, an ANN is computationally relatively cheap, compared to a large collection of GMMs: A single ANN computation produces the likelihoods for all phoneme classes in a given frame.

Through the late 1980s and 1990s, results reported for hybrid models on tasks such as Wall Street Journal were competitive with conventional HMMs using GMMs (e.g., [62]). These systems subsequently led to the tandem approach (see Section 4.4) which combines the advantages of both hybrid and conventional generative approaches; this is used in some current systems to provide very good results on the latest tasks, like Fisher. Although it appears that tandem systems always outperforms hybrid ones, the latter is simpler to implement and was therefore the first approach tried in the workshop.

4.3.1 Hybrid phone-based models

In most previous work, the classes used are phonemes (or perhaps context-dependent phonemes, but we will not consider this case further). When used in conjunction with HMMs, the system has one HMM per phone. The number of states in each HMM is used simply as a minimum duration constraint. The observation likelihood for all of the states in a particular phoneme is simply read off the corresponding ANN output (normalized, divided by the prior).

4.3.2 Hybrid AF-based models

In our experiments with AF-based MLPs, the sub-word unit used in the pronunciation dictionary was always the phoneme; phonemes are then mapped to AFs (possibly with a context-dependent and/or learned mapping in some of our models). Using AF-classifying MLPs introduces a problem: There is not a simple correspondence between ANN outputs and the back-end HMMs-of-phonemes. How then, do we compute the state emission likelihood, given the multiple AF-classifying ANN outputs?

Our solution was to build models, shown in Figure 4.1 incorporating explicit hidden variables for the AFs, which were mapped from the phone variable in one of two ways, deterministic or probabilistic. The ANNs then provide so-called "virtual evidence" [72] for the AF variables.

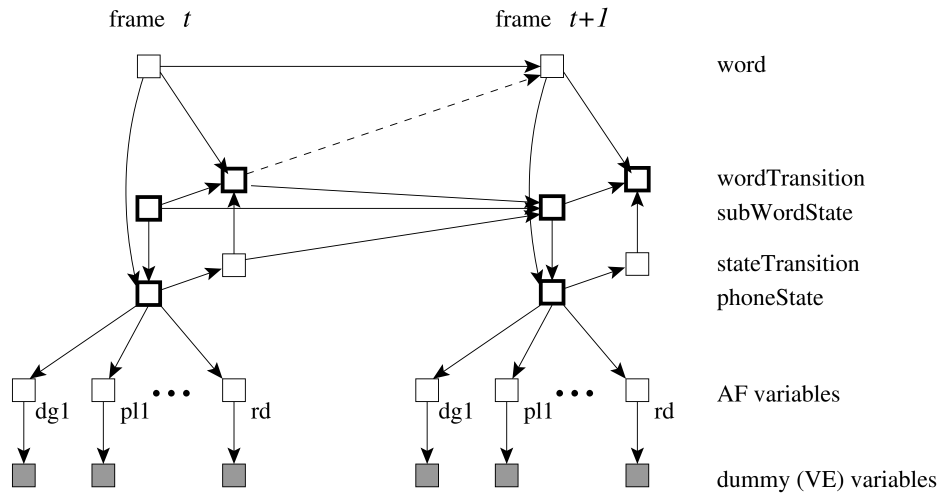


Figure 4.1: Hybrid model, using virtual evidence provided by AF classifiers.

4.3.3 Experiments

Deterministic mapping from phone to AFs

The AF targets used to train the ANNs were derived using a deterministic mapping from phone labels; each phone maps to a unique AF configuration. If this initial phone labelling is perfect, the phoneme-to-AF mapping is a true representation of the shared feature values between phonemes, and the ANNs perform AF classification with high accuracy, then a recognizer using a deterministic mapping should perform well. This mapping sets each AF to the *canonical* value, given the phone value. From the results in Table 4.2, however, we can see that this model performs very poorly: a small CV set WER of 32.9%.

Probabilistic dependence of AFs on the phone

In the next model we studied, dense conditional probability tables (CPTs) are used for the relationship between each AF and the phone variable. Training this model for the 500-word task was somewhat more involved, because initializing these dense CPTs with uniform probabilities leads to a computationally expensive model. The first strategy adopted was to train on a 1000-utterance subset of the training data,

10 word task		
	WER	
model	**small CV set**	**test set**
Hybrid monophone, det. phone-to-AF mapping	32.9	-
Hybrid monophone, non-det. phone-to-AF mapping	26.0	30.1
Hybrid monophone, non-det. phone-to-AF mapping, embedded training	23.1	24.3
Hybrid monophone, non-det. phone-to-AF mapping, plus PLPs	16.2	19.6

500 word task		
	WER	
model	**small CV set**	**test set**
Hybrid monophone, non-det. phone-to-AF mapping	66.6	-

Table 4.2: Word error rates for the hybrid models.

then to make the CPTs sparse by zeroing all entries smaller than 0.1 and renormalizing. At this point, the DBN structure was re-triangulated, to find a triangulation that was faster for the particular sparsity pattern chosen. Then, training to convergence was completed on the full training set. In subsequent experiments, we found triangulations using a genetic algorithm [3], which allowed the uniformly-initialized CPT model to be trained directly on the full training set. However, in order to decode in reasonable amounts of memory, the CPTs still had to be made sparse (again, by zeroing all entries smaller than 0.1 and renormalizing).

This model performed much better than the deterministic one, achieving a small CV set WER of 26.0% compared to 32.9% (Table 4.2).

Embedded training

Thus far, the ANNs used to estimate the AF posteriors had only been trained on data with canonical AF labels derived from phone alignments. Once a working AF-based hybrid model was available, it was possible to re-label the training set directly with AF labels, using Viterbi decoding of the AF random variables in the hybrid model. This produces a forced alignment at the AF level. The ANNs were then re-trained (on only the SVB data) using these alignments. Then, given the new set of posteriors produced by these re-trained ANNs, the hybrid model itself was retrained. As a result, the WER was reduced from 26.0% to 23.1% (Table 4.2).

Hybrid models, plus acoustic observations

In order to combine the benefits of the hybrid model (i.e., its use of discriminatively-trained AF classifiers) with those of conventional models using Gaussians, we added a PLP observation variable to the hybrid model, as shown in Figure 4.2. The PLP observations have a Gaussian mixture distribution as in a standard HMM-based recognizer, while the rest of the model is as described above for the models using only the MLP outputs. This model had the lowest overall WER of all of the hybrid models, 16.2% (Table 4.2), which is marginally better than the conventional monophone HMM baseline (16.7% on the 10-word task small CV set).

4.3.4 Adding dependencies to hybrid models

We also studied whether our models could benefit from additional learned dependencies. Our approach was to make simple localized changes to the model and then test the model performance. We applied the "explaining away residue" (EAR) first proposed by Bilmes [7] as an information-theoretic criterion for structure learning in DBNs.

The EAR measure between two random variables is defined as

$$EAR(X,Y) = I(X,Y|Q) - I(X,Y) \tag{4.1}$$

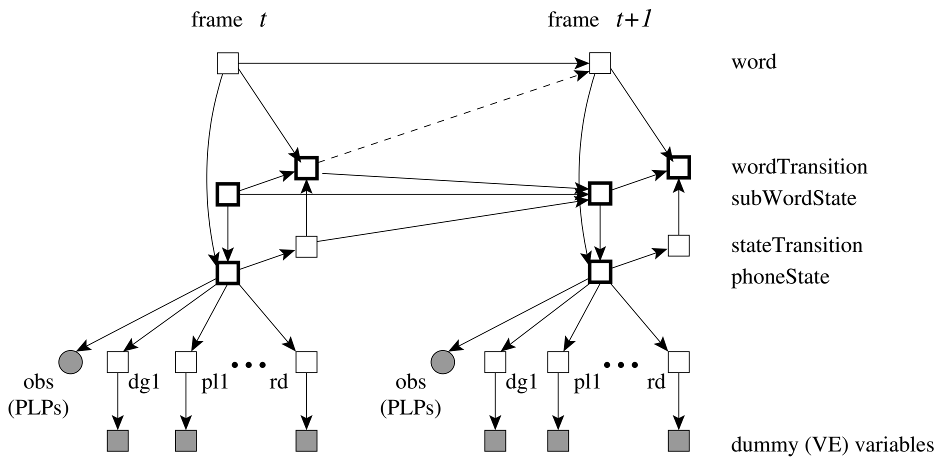


Figure 4.2: Hybrid model, using virtual evidence provided from an AF classifier, plus a PLP observation modelled using a GMM.

where I represents mutual information, and Q represents the class value. This quantity attempts to measure the added discriminative power of having an edge between X and Y. For our investigation, we tried several different possible class types for Q. Ideally Q would be the vocabulary and this measure would help us determine if an edge improves our ability to discriminate between words. Another reason to use words as the class is that while we have canonical frame-level word labels, no such data exists for phones.

Computing mutual information $I(X, Y)$ requires estimates for the probability distributions $P(X)$, $P(Y)$, and $P(X, Y)$. One approach to estimating these was to examine the values of the random variables after Viterbi decoding and binning these values to create a distribution. Another was to use GMTK to determine the estimated distributions after decoding. Since all of the variables involved are discrete, the distributions consist of tables and can be enumerated.

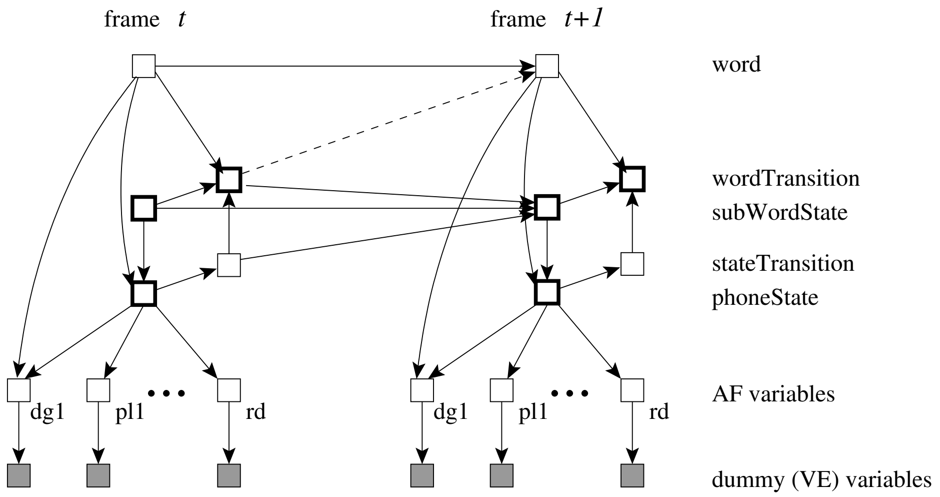


Figure 4.3: Monophone hybrid model, augmented with a word-dg1 dependency.

The model tested was a simple monophone hybrid, and we attempted to determine whether adding a direct dependency between the word variable and an articulatory feature variable would make a difference. Given the eight articulatory feature variables, we chose edges to add in two different ways: using either the EAR measure or the word error rate on the cross validation set. The models were trained and tested for the 10-word SVitchboard task.

As can be seen in Table 4.3.4, the performance improvements are modest. The ROU+VOW combination

Edge added (from Word)	CV WER (%)	Test WER (%)
vowel	25.7	29.5
frontness	26.5	29.5
rounding	25.1	29.7
place	26.4	29.8
height	27.1	29.8
baseline (no added edges)	26.0	30.0
nasality	26.2	30.0
glottal state	26.0	30.1
degree/manner	26.9	30.3
rounding + vowel	25.1	29.9

Table 4.3: Performance of augmented hybrid models on the 10-word SVitchboard task, in order of decreasing WER.

was chosen since those two edges had the best performance on the cross-validation data. If we attempt to correlate the performance with the EAR measure, we find that the word-feature pair with the best EAR measure was the GLO random variable; the model performed slightly worse on test data with this edge than without it. In short, we can say that we discovered no large improvements to be had with simple changes to our models. Although the observed changes did not correlate well with the EAR measure, the changes were also modest overall; we can therefore make no definitive statement about the efficacy of the measure in this case.

4.4 Tandem models

In this section, we present our development and investigations of an AF-based tandem model.[2] The so-called *tandem* approach, where the posteriors of a multilayer perceptron (MLP) classifier are used as observations in an ASR system, has proven to be a very effective method. Most tandem approaches to date rely on MLPs trained for phone classification, and append the post-processed posteriors to some standard observation vector in an HMM-based system. In the workshop, we developed an alternative tandem approach based on MLPs trained for articulatory feature classification. We also developed a *factored* observation model for modeling separately the posterior-based and standard observations, allowing for separate hidden mixture and state-tying structures for each factor.

4.4.1 Introduction

The *tandem* approach refers to a data-driven signal processing method for extracting observations for acoustic modeling [41, 22, 91]. The tandem approach involves first training a MLP to perform phone classification at the frame level, and then using the post-processed frame-level phone posterior estimates of the MLP as the acoustic observations in HMMs. Commonly, the tandem features are appended to some standard feature vector such as the perceptual linear prediction (PLP) coefficients. The tandem approach has given significant performance improvements for large-vocabulary speech recognition of English conversational telephone speech (CTS) [91], and Arabic and Mandarin broadcast news and conversations [52, 90]. The tandem approach also seems to have some cross-domain and cross-language generalization ability [81].

The majority of previous tandem approaches have been limited to phone-based posterior estimation. In this workshop, we proposed an alternative approach based on articulatory features. The articulatory feature-based tandem observations are extracted from the posterior estimates of a set of MLPs trained for AF classification, one for each feature. One of the main motivations for the proposed approach, as opposed to the traditional phone-based tandem approach, is that AF classification is simpler, involving multiple classification problems with a small number of classes each, instead of a single phone classifier with a large number of classes. In addition, while not explored in this work, AFs are more language-universal than

[2]Parts of this section have appeared in [15].

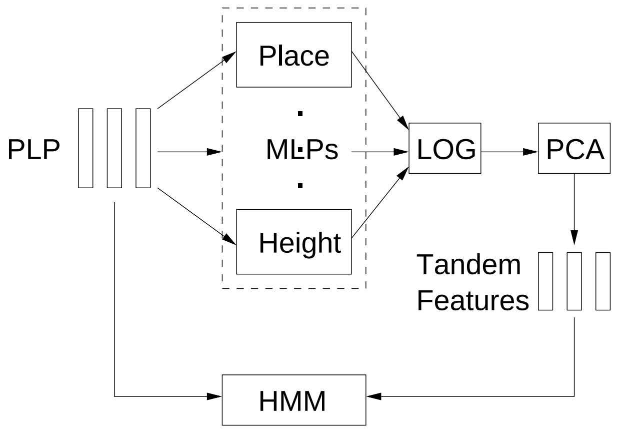


Figure 4.4: Articulatory feature-based tandem processing.

phones, and therefore they can better generalize to new languages [82]. A future goal is therefore to apply the AF-based tandem approach to cross-linguistic settings.

As described above, the usual approach in tandem systems is to concatenate the MLP posterior-based observations with some standard observation vector, and then to model the concatenated vector with a single HMM output distribution. The standard and posterior vectors are forced to have the same hidden mixture and state-tying structure. However, the two types of observation vectors are likely to have quite different statistical properties, being derived from two different paradigms, prior knowledge/heuristics vs. data-driven learning [81]. The large dimensionality of the concatenated vector also gives highly concentrated Gaussian probability estimates. In the workshop, we developed an alternative, factored modeling approach, where each of the observation vectors is modeled separately, avoiding these problems. Our approach is similar to [49], which also proposed an AF-based tandem method, but we directly combine the posteriors with the standard observations at the HMM outputs, and experiment with new observation models.

4.4.2 Articulatory feature-based tandem observations

Tandem processing

We extract the AF-based tandem observations as follows (see Figure 4.4). For each time frame, the posterior probability estimates from the each of the eight AF MLPs are concatenated, producing a 64-dimensional vector. Their logarithm is taken to expand the dynamic range of the posterior probabilities to the real axis. Roughly speaking, the logarithm undoes the effect of the final softmax nonlinearity at the MLP outputs, making them more amenable to modeling with Gaussian mixtures. Next, principal component analysis (PCA) is applied to reduce the dimensionality to 26, which was determined to contain the 95% of the total variance. The PCA eliminates redundancies among the posterior probabilities from different MLPs; such redundancies often exist since the AF set that we use is not orthogonal. The PCA transform is estimated on the MLP training set. Finally, per-speaker mean subtraction and variance normalization are applied. The resulting 26-dimensional vectors, along with the standard 39-dimensional PLP vectors, are used as acoustic observations in a phone HMM-based recognizer.

Experiments

Using the procedure in Figure 4.4, we extracted two sets of 26-dimensional AF-based tandem observation vectors, one using the MLPs trained on Fisher and another using those trained on SVitchboard. For comparison, we also generated a set of 26-dimensional phone-based tandem observations using the phone MLP trained on SVitchboard. In the first four lines of Table 4.4, we report the WERs for the baseline monophone system using PLPs, and for monophone systems using tandem observations from different MLPs: Fisher AF MLPs, SVitchboard AF MLPs, and the SVitchboard phone MLP. In the first two lines of Table 4.5, we report the WERs for within-word triphone systems using PLPs and using the concatenated PLPs and the tandem observations from the Fisher AF MLPs. The number of components per Gaussian mixture was 128 for the monophone systems and 64 for the triphone systems.

Model	WER (%)
PLP	67.7
Phone tandem (Fisher)	61.4
AF tandem (Fisher)	61.1
PLP + phone tandem (SVB)	63.0
PLP + AF tandem (SVB)	62.3
PLP + phone tandem (Fisher)	58.2
PLP + AF tandem (Fisher)	59.7
PLP + AF tandem (Fisher) + Factoring	59.1

Table 4.4: WERs of the monophone systems using PLPs and of various systems using the posterior-based observations by themselves and in combination with PLPs, on the 500-word test (E) set. The corpus name in parentheses refers to the MLP training set. The tandem systems in lines 4–7 used concatenated PLP and posterior-based observations, and "factoring" refers to a factored observation model (see Section 4.4.3).

A few observations about the results of Tables 4.4 and 4.5 are in order. First, all of the systems using any type of tandem observations (phone or AF-based, SVitchboard- or Fisher-trained) in both monophone and triphone models significantly improve the performance over the baseline PLP system. Systems using only the tandem observations themselves significantly outperform those using only PLPs, while their combination brings a further gain. Therefore, PLPs and tandem observations are complementary. Second, the AF-based tandem observations are as effective as the phone-based ones when the MLP training is done on the SVitchboard training data set, which is small (the difference is not statistically significant). However, when the MLP training is done on the Fisher data set, the system using the phone tandem observations performs somewhat better in monophone modeling and vice versa in triphone modeling. Third, the Fisher MLPs are significantly better than the SVitchboard MLPs for the purposes of AF-based tandem processing. This is expected given that the Fisher MLPs were trained on two orders of magnitude more training data. Fourth, the AF-based tandem observations are also very effective in triphone modeling, even though the relative improvement is lower (12% for monophones vs. 7% for triphones). All pairs of results in Tables 4.4 and 4.5, except the phone- vs. AF-based tandem pair in Table 4.4, are statistically significant according to the matched pairs sentence-segment word error test ($p < 0.01$) [29]. effective.

4.4.3 Factored tandem observation models

Factored model

In the basic tandem approach described in Section 4.4.2 and also in most previous work (e.g., [41, 49, 22, 91]), the tandem observation vectors have simply been concatenated with PLPs and then jointly modeled using the same hidden mixture and state-tying structures in HMMs with diagonal-covariance mixture distributions. The HMM output distributions of these concatenated observations can be represented as

$$p(x, y|q) = \sum_t p(t|q)\, p(x|t, q)\, p(y|t, q) \tag{4.2}$$

where x and y denote the PLP and tandem vectors, respectively, q denotes the HMM state, and t denotes the hidden mixture component. See Figure 4.5(a) for a graphical model depiction (there is no arrow between x and y due to the diagonal-covariance assumption).

The PLP and tandem vectors are generated in two very different ways, prior knowledge and heuristics vs. data-driven learning. Their statistical characteristics are likely to be quite different, and enforcing the same mixture structure and decision-tree state-tying scheme could be inefficient for learning their distributions from sparse data. In addition, the large dimensionality of the concatenated vector gives highly concentrated Gaussians, which in previous work has been dealt with using a heuristic weighting factor [91]. Instead, here we propose a principled approach based on factored modeling of the PLP and tandem observation vectors at the HMM output distributions, allowing for separate hidden mixture and state-tying

Model	**# of states**	**WER (%)**
PLP	675	61.7
PLP + phone tandem (Fisher) concatenated	441	54.9
PLP + AF tandem (Fisher) concatenated	426	55.4
PLP + AF tandem (Fisher) factored	689 / 302	54.4
PLP + AF tandem (Fisher) factored + tied state	426 / 426	54.9

Table 4.5: WERs for the various triphone systems. The number of states refers to the number of decision-tree clustered triphone states; the pair for the observation factored model is the number of states for the PLP and the tandem, respectively, factors. See Table 4.4 caption for the notation.

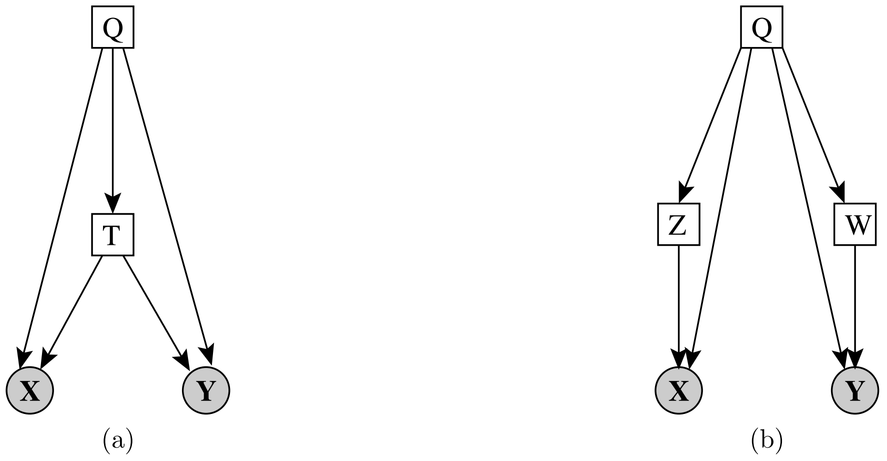


Figure 4.5: (a) Feature concatenation (assuming diagonal covariance Gaussian modeling), and (b) factored modeling.

structures for each vector. In particular, the HMM output distributions for the PLP and tandem vectors are factored as follows, as in multi-stream ASR models [11]:

$$\begin{aligned} p(x, y|q) &= \Big(\sum_z p(z|q)\, p(x|z, q)\Big) \\ &\quad \times \Big(\sum_w p(w|q)\, p(y|w, q)\Big) \end{aligned} \tag{4.3}$$

where w and z are the hidden mixture variables for x and y, respectively. See Figure 4.5(b) for a graphical model depiction. The product of sums in Equation 4.3 has a smoothing effect.

Notice that in Equation 4.3 the PLP and tandem vectors are assumed to be conditionally independent given the HMM state, whereas in Equation 4.2, they are indirectly coupled via the hidden mixture variable t. However, in general, neither form is subsumed by the other one due to Gaussian parameterization, and there are distributions that can be represented by one and not the other one, and vice versa. The factored modeling has the advantage that it can more accurately characterize each of the PLP and tandem observation vectors by modeling each of them separately. On the other hand, it could be at a disadvantage if the PLP and tandem features are highly correlated even when conditioned on the HMM state.

Fully factored model

We can factor the MLPs even further by assuming that all of the MLP outputs are conditionally independent given the HMM state. Letting y_j be the suitably processed output of the j^{th} MLP, the observation model becomes

$$p(x, y_1, ..., y_j, ..., y_8|q) = \Big(\sum_z p(z|q)\, p(x|z, q)\Big) \prod_{j=1}^{8} \Big(\sum_{w_j} p(w_j|q)\, p(y_j|w_j, q)\Big) \tag{4.4}$$

The outputs y_j are obtained similarly to the procedure in Figure 4.4, except that the PCA is applied to each log-MLP output individually. The eigenvectors accounting for 95% variance *within each* MLP are kept. In this case the total number of dimensions is greater than in the previous model. Table 4.6 shows the number of dimensions contributed by each MLP.

MLP	**# dims before PCA**	**# dims after PCA**
place	10	7
degree/manner	6	4
nasality	3	2
glottal state	4	2
rounding	3	2
vowel	23	13
height	8	5
frontness	7	5

Table 4.6: The number of dimensions contributed by each MLP after keeping only the eigenvectors of the PCA responsible for the 95% of the variance of each MLP. The total number of dimensions is 40, which is greater than the 26 dimensions obtained by taking the PCA of all MLPs together as in the previous section.

It may happen that some factor y_j is a better predictor of the hidden phoneme q, and so it makes sense to weight the factors so as to optimize the WER. The observation model becomes

$$p(x, y_1, ..., y_j, ..., y_8|q) = \Big(\sum_z p(z|q)\, p(x|z, q)\Big) \prod_{j=1}^{8} \Big(\sum_{w_j} p(w_j|q)\, p(y_j|w_j, q)\Big)^{\lambda_j} \tag{4.5}$$

where λ_j is the weight of the j^{th} stream. The selection of λ_j is an optimization problem in its own right. In addition to optimizing the 8 factor weights, we simultaneously optimized the language model penalty and language model scale parameters. We have chosen to optimize all 10 parameters using the amoeba algorithm [65]. It searches for a local minimum greedily and does not depend on the derivative of the objective function. The fact that the algorithm does not make use of the gradient of the objective function was desirable in this case because we expected the objective function to be non-smooth. Experiments demonstrated that the WER is in fact not smooth as a function of the language model penalty and language model scale parameters.

Experiments

Using the AF-based tandem observations from the Fisher MLPs, we have compared factored modeling to concatenation of the observation vectors. The results with monophone models are in Table 4.4, and those with triphone models are in Table 4.5. The decision tree-based state clustering for the triphone system with the factored observation model was performed exactly the same way it was performed for the feature-concatenated system, except that the clustering procedure was invoked twice, one for each factor in Equation 4.3.

We can draw the following conclusions from Tables 4.4 and 4.5. First, factored observation modeling significantly improves over feature concatenation (differences are statistically significant). Therefore, the benefit from the more accurate characterization of the PLP and tandem vectors individually seems to outweigh the loss from the conditional independence assumption (and/or, the dependencies are weak). Second, the clustering results demonstrate that the factored approach not only gives better results, but it is also more parsimonious. The system with factored observation models has about 40% fewer parameters. The decision trees for the PLP and tandem observations differ in both structure and size, corroborating the intuition that the statistical properties of the PLPs and tandem features are quite different. As we observe in the last line of Table 4.5, enforcing the same decision tree structure for the PLP and tandem features results in slightly poorer recognition performance.

Chapter 5

Analyses using manual articulatory feature transcriptions

We have previously remarked that one of the obstacles for articulatory feature-based recognition research is the lack of "ground truth" labels, for both training and testing of AF classifiers. We began to address the problem of inadequate training labels with embedded training (Section 4.3.3); this improved recognition performance, but of course did not provide what a human might call acceptable "ground truth". For testing classifiers and alignments, it is useful to have a small set of careful manual labels. We have begun to address this by collecting a set of manual feature-level transcriptions of Switchboard utterances.[1]. It was not our goal to transcribe sufficient data for *training* of classifiers or recognizers; this was not considered to be feasible within a reasonable time, and we believe that embedded training is a more viable approach for obtaining training labels. Instead, we focused on obtaining a small, carefully labeled set that is large enough for *testing* of classifier outputs or alignments. These analyses are presented in Sections 5.2 and 5.3.

A major concern is whether this type of labeling can be done reliably and with reasonable speed. We therefore begin, in Section 5.1, by describing in detail the data collection methods and analysis of the data for inter-transcriber agreement and prevalence of canonical vs. non-canonical labels. One question is whether we can obtain a more detailed transcription at the feature level than would be afforded by, for example, converting existing phonetic transcriptions such as those of the Switchboard Transcription Project (STP, [37]), while maintaining reasonable transcriber agreement.

5.1 Data collection

The main data set consists of 78 utterances drawn from SVitchboard [45]. Of these, 33 were hand-selected for phonetic variety and 45 were randomly drawn from all 5- to 14-word SVitchboard utterances.[2] An additional 9 utterances drawn from the STP data set were also transcribed, for comparisons of the two types of utterances.

The utterances were labeled by two transcribers, a phonetician and a graduate student in a speech research group.[3] The transcribers did not receive formal training for the labeling task, but rather participated in the refinement of the feature set and transcription interface while labeling practice utterances. After several weeks of practice, the transcription interface and guidelines were fixed; all of the utterances described here (and provided for download) were transcribed after this practice period.

The transcription interface was based on KTH's WaveSurfer [84], a highly configurable sound analysis and annotation tool. Figure 5.1 shows a screen shot of a completed transcription. Besides the feature tiers, the transcribers were provided wide-band and narrow-band spectrograms and a plot of the signal

[1]The manual transcriptions are available for download from `http://people.csail.mit.edu/klivescu/twiki/bin/view/WS06`. Parts of this section have appeared previously in [55].

[2]The lower limit on utterance length was chosen to avoid the most common filler-type utterances such as "Yeah okay" and "Good talking to you"; the upper limit was chosen to avoid transcriber fatigue and ensure a greater variety of utterances.

[3]Lisa Lavoie and Xuemin Chi.

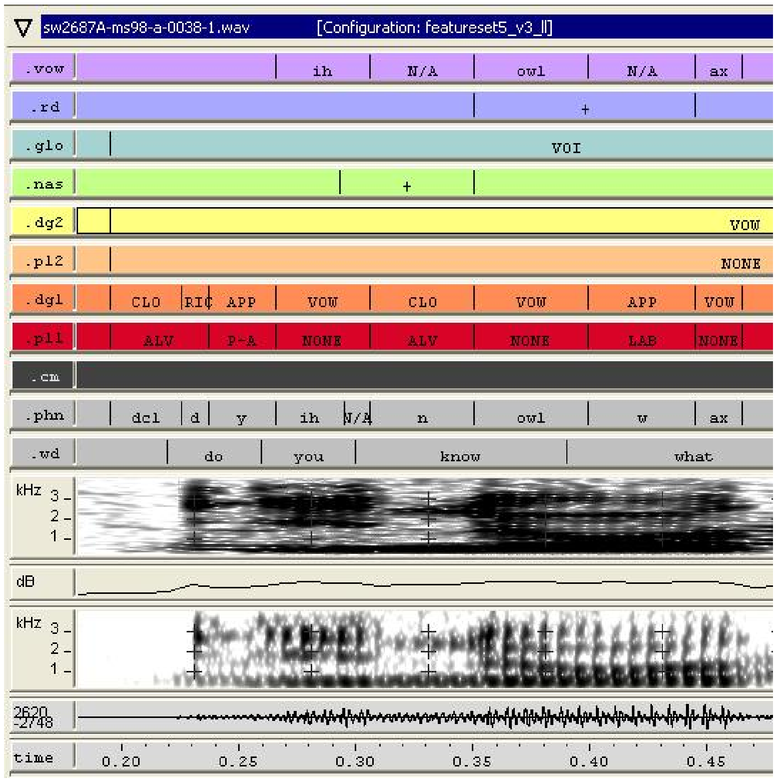


Figure 5.1: Screen shot of the transcription interface.

power. There was no time restriction and the transcribers could use all sources of information available in WaveSurfer (e.g. listening to arbitrary waveform segments, generating spectral slices or waveform blow-ups, and modifying the spectrogram parameters). The procedure developed for transcription consisted of three passes, designed to minimize transcriber fatigue and error. The procedure was as follows:

- *Initialization.* Transcribers were provided initial word and phone alignments, which could be modified during transcription. The word alignments were the same Mississippi State University alignments used in our recognition experiments [27]. The phone alignment was a manual alignment (done by Karen Livescu), within the given word boundaries, to a *dictionary* pronunciation of each word.[4] The initial and final silence feature values were filled in automatically based on the word alignments, but could also be modified by the transcribers.

- *First pass.* Each transcriber labeled the utterances using a hybrid phone-feature labeling: For any segment that could be described as a canonical phone, the phone label was used; otherwise, the feature tiers were used. Figure 5.2 shows a portion of a first-pass transcription. These transcriptions were then automatically converted to all-feature transcriptions before proceeding. From this point on the phone tier was dropped and only the feature tiers were used. The purpose of this hybrid labeling approach was to increase productivity and avoid transcriber fatigue and error.

- *Second pass.* Each transcriber compared her transcriptions to the other transcriber's and corrected any errors in her own labeling (but not disagreements). Each transcriber performed this pass independently, without discussion among transcribers. For this purpose, a different WaveSurfer configuration was used, with the two transcriptions viewed together; see Figure 5.3.

- *Third pass.* Transcribers discussed disagreements and, possibly, modified their transcriptions.

[4]Based on a dictionary from the MIT Spoken Language Systems Group, also the same as the dictionary used in our recognition experiments.

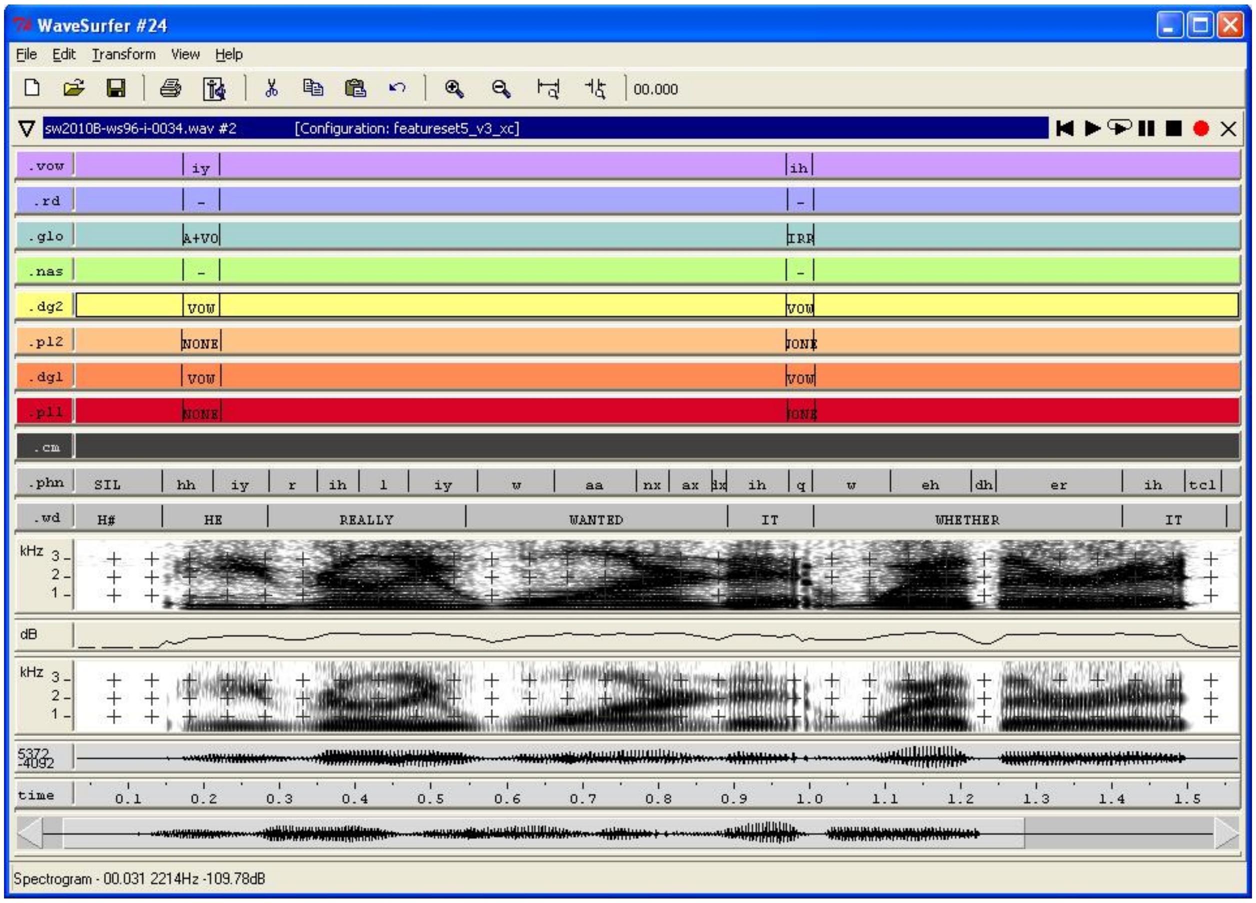

WaveSurfer #24
File Edit Transform View Help
00.000
sw2010B-ws96-i-0034.wav #2 [Configuration: featureset5_v3_xc]
.vow iy ih
.rd - -
.glo A+VO IRR
.nas - -
.dg2 VOW VOW
.pl2 NONE NONE
.dg1 VOW VOW
.pl1 NONE NONE
.cm
.phn SIL hh iy r ih l iy w aa nx ax dx ih q w eh dh er ih tcl
.wd H# HE REALLY WANTED IT WHETHER IT
kHz 3 2 1
dB
5372 -4092
time 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5
Spectrogram - 00.031 2214Hz -109.78dB

Figure 5.2: Screen shot of a first-pass phone-feature hybrid transcription.

This procedure was followed in batches of roughly 15 utterances at a time, each of which were transcribed in the span of about a week. The transcribers followed a set of labeling conventions and guidelines.[5] Some aspects of articulation are, of course, not discernible from the signal (such as a tongue constriction during a [p] closure); however, only those aspects that are acoustically salient need be labeled correctly. Another way to view this is that the transcriptions should reflect what we would like automatic AF classifiers to be able to detect.

Averaged over the 78 SVitchboard utterances excluding initial and final silences (a total of 119s of speech), the speed of transcription was 623 times real-time for the first pass and 335 times real-time for the second pass.

The nine STP utterances were treated somewhat differently. In order to analyze the effect of using the hybrid phone-feature format in the first pass, four of the STP utterances (9.9s of non-silence speech) were transcribed as described above, and the remaining five (11.4s) were transcribed using an all-feature format in the first pass. We found that, for these utterances, the first-pass real-time factor was 328 for the phone-feature hybrid transcriptions and 799 for the all-feature transcriptions.

5.1.1 Analysis of data collection

Inter-transcriber agreement

To measure the reliability and usefulness of our approach, we examined both inter-transcriber agreement and the "canonicalness" of the labels (in a sense described below). Since the transcribers were independently

[5]Available at `http://people.csail.mit.edu/klivescu/twiki/bin/view.cgi/WS06/TranscriptionNotes`

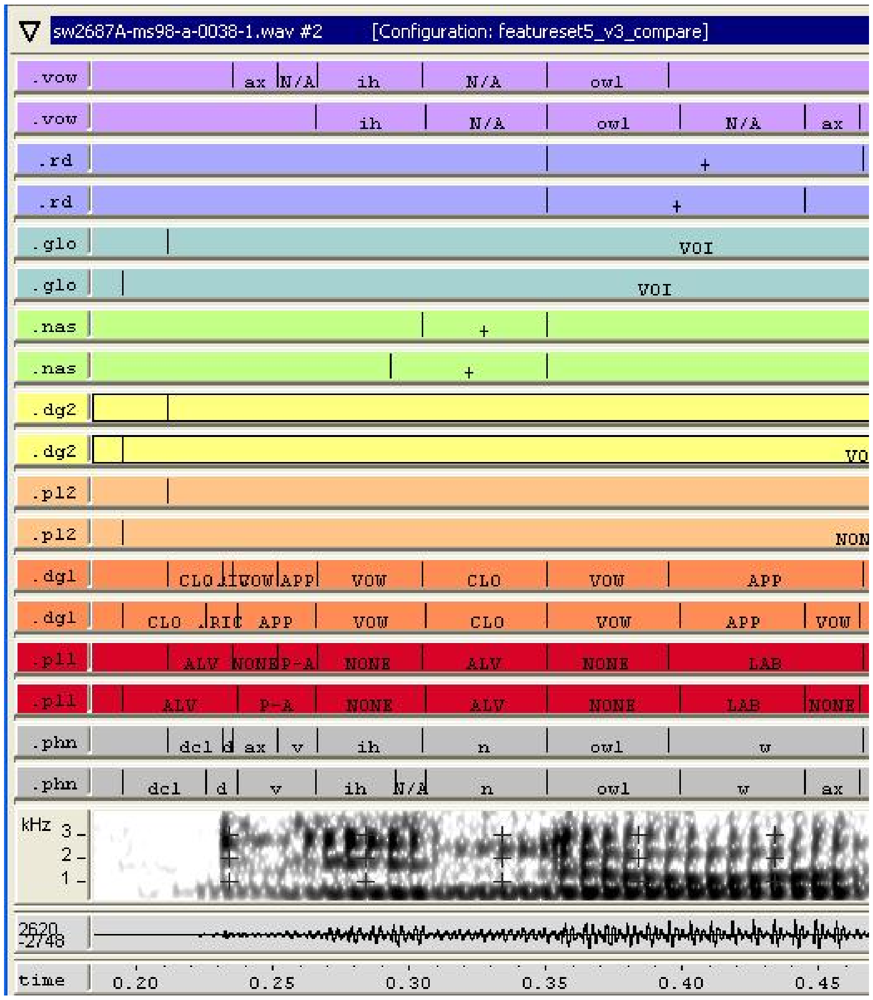

sw2687A-ms98-a-0038-1.wav #2
[Configuration: featureset5_v3_compare]
.vow
.rd
.glo
.nas
.dg2
.pl2
.dg1
.pl1
.phn
kHz
time
0.20
0.25
0.30
0.35
0.40
0.45

Figure 5.3: Screen shot of the transcription interface for second-pass transcriptions.

labeling both the features and the corresponding time boundaries, we take two different approaches when computing their agreement. The simplest approach is to discard all of the timing information and calculate the string edit distance between the two transcribers' feature labels. The "label agreement" is then 1 minus the string edit distance. We can also compute Cohen's unweighted kappa statistic to account for chance agreement [19],

$$k = \frac{P_{\text{obs}} - P_{\text{exp}}}{1 - P_{\text{exp}}}, \tag{5.1}$$

where P_{obs} is the label agreement and P_{exp} is the proportion of labels expected to be in agreement due to chance. Here, P_{exp} is calculated by subtracting from one the string edit distance between the concatenation of both transcripts and a string of the same length consisting of repetitions of the most frequently occurring label. The kappa statistic is equal to one for complete agreement and does not exceed zero if the observed agreement is no greater than chance. Landis and Koch [50] provide a guide (reproduced in Table 5.1.1) to approximate the strength of agreement for a given kappa statistic value.

Our second approach is to calculate the amount of *time* that the two transcribers are in agreement and the corresponding time-weighted kappa statistic:

$$k_{tw} = \frac{t_{\text{obs}} - t_{\text{exp}}}{t_{\text{total}} - t_{\text{exp}}}, \tag{5.2}$$

Kappa	**Strength**
$k < 0.2$	Poor
$0.2 < k \leqslant 0.4$	Fair
$0.4 < k \leqslant 0.6$	Moderate
$0.6 < k \leqslant 0.8$	Good
$0.8 < k \leqslant 1$	Very good

Table 5.1: Strength of agreement corresponding to different ranges of the kappa statistic.

where t_{obs} is the observed time in agreement, t_{exp} is the expected time in agreement, and t_{total} is the total time. As seen in Figure 5.4, the time-weighted kappa statistic is indicative of very good agreement for all of the features except nasal, rounding, and vowel. The first two most likely have a lower kappa statistic because of the very high chance agreement for these two binary features. For the vowel feature, the decreased agreement may be due to the large number of values for the vowel feature and the correspondingly increased difficulty of classification. When examining the unweighted kappa statistic, only three of the eight features showed very good agreement, although all showed good agreement. This is most likely due to the effect of short spans of disagreement in the transcript being given the same weight as longer spans. Although this approach is used when calculating word error rates, it seems less applicable for time-aligned transcriptions. Overall, both kappa statistics show very good agreement between the two transcribers.

Figure 5.5 shows the inter-transcriber confusion matrices for the degree and place of the foremost constriction in terms of time-weighted agreement. These figures indicate that Transcriber 1 was more likely than Transcriber 2 to use the "vowel" manner label (and, correspondingly, the "none" place label). For the remaining figures, we consider only the time-weighted kappa statistic.

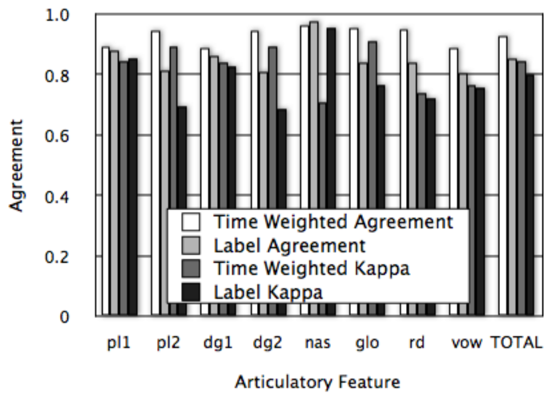
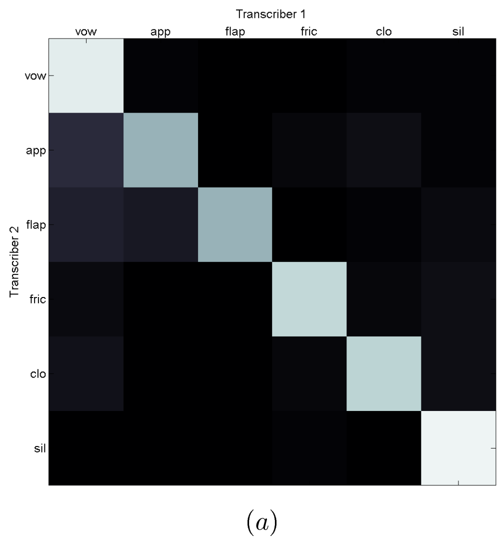


Figure 5.4: Transcriber agreement metrics on 78 SVitchboard utterances.

Because of the multi-pass transcription strategy, there is a concern that agreement between the transcribers may be due not to real agreement in their opinions, but to "convergence" after viewing each other's transcriptions in the second and third passes. To determine whether this is the case, we compare agreement across the multiple passes (Figure 5.6). While the agreement usually improves with each successive pass, there is not a dramatic difference, with the same five features as before in the "very good" agreement range.

We also attempted to compare the inter-transcriber agreement on the newly collected data to that of STP's transcribers. This is a difficult task for two reasons. First, the two transcription systems use different label sets. To compute agreement with respect to the same labels, we first converted the STP phonetic labels to our feature set. Second, the 9 STP utterances we chose to transcribe were not all transcribed by multiple STP labelers. STP provides agreement data (i.e. multiple transcriptions of the same utterances) for 11 utterances, of which only one is contained in our set of 9. For this reason, the agreement measures we provide should be viewed as only a rough guideline of the overall agreement differences. Figure 5.7 shows the time-weighted kappa statistic separately for three sets of utterances: the 4 transcribed with AF labels using our "standard" procedure (with a phone-feature hybrid first pass); the 5 transcribed with AF labels



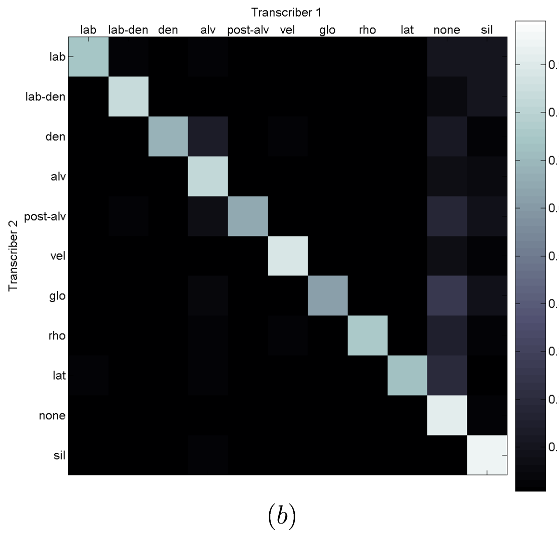


Figure 5.5: Inter-transcriber confusion matrices for **dg1** (a) and **pl1** (b) in the 78 manually transcribed SVitchboard utterances.

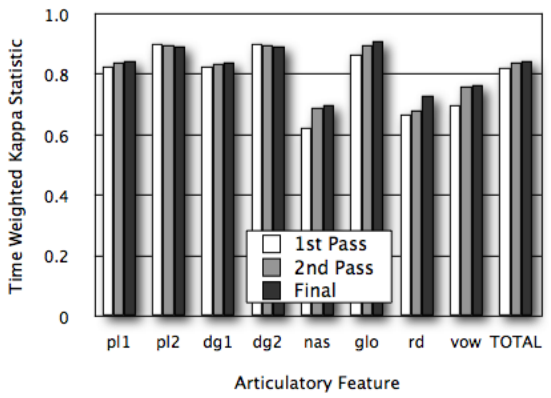


Figure 5.6: Time-weighted kappa statistic for each of the three transcription passes.

but using features only in all passes; and the 11 provided in the STP agreement data.

This figure suggests that the agreement among WS06 transcribers is higher than among the STP transcribers. This must be viewed as a tentative conclusion, due to the different label sets, the different sets of utterances, and the small amount of data overall. However, the high level of agreement between the WS06 transcribers is very encouraging.

"Canonicalness" of transcriptions

Another question of interest is the extent to which the transcribers used the extra flexibility of the feature tiers that is not available in a phonetic transcription. In other words, how much of the speech could have been faithfully represented with canonical phone labels? Table 5.2 shows the percentage of the time, excluding silences, for which the transcribers used canonical feature configurations. For this purpose, a "canonical feature configuration" is defined as any vector of feature values that corresponds to one of the dictionary phones. For the SVitchboard utterances, the transcribers used non-canonical labels 12-14% of the time. For the STP utterances, there was a wider range. The results indicate that the WS06 transcribers were more likely than STP's transcribers to use non-canonical feature configurations when using an all-feature format in the first pass, and less likely to do so when using a phone-feature hybrid format in the first pass. This

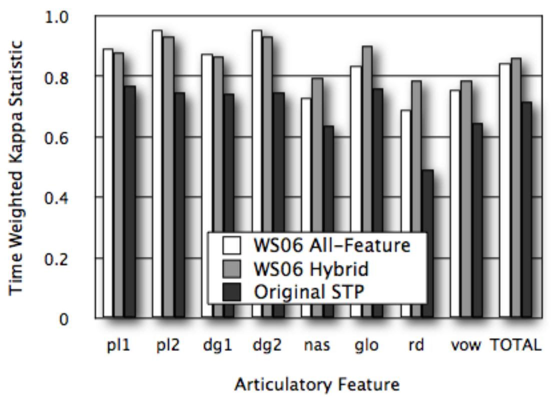


Figure 5.7: Time-weighted kappa statistic for transcriptions of STP utterances by WS06 and STP transcribers. The WS06 transcriptions have been split into the 5 utterance that had all-feature first-pass transcriptions and the 4 utterances with hybrid phone-feature first-pass transcriptions.

fact, combined with the high inter-transcriber agreement even when using the all-feature format, implies that future transcription efforts may benefit from investing the added time to do all-feature transcriptions, or to otherwise improve the transcription interface to encourage greater use of the feature tiers.

There are many other questions we may wish to ask about the transcriptions, such as what are the common patterns of non-canonical feature values. The analysis we have performed thus far is, we believe, sufficient to establish the reliability and usefulness of the manual transcriptions. We are therefore now in a position to make some measurements of automatic classifier and alignment accuracy against this human-determined "ground truth".

Set	**Transcriber**	**% canonical**
STP all-feat (11.4s)	WS06 transcriber 1	81.8
	WS06 transcriber 2	73.5
	STP	84.7
STP hybrid (9.9s)	WS06 transcriber 1	95.3
	WS06 transcriber 2	95.4
	STP	91.6
SVitchboard (112.8s)	WS06 transcriber 1	86.2
	WS06 transcriber 2	88.4

Table 5.2: Percent of the time for which canonical feature configurations were used, for different utterance sets and transcribers. In the leftmost column, numbers in parentheses represent the total (non-silence) duration of the utterance set.

5.2 Analysis of AF classifier performance

In this section we analyze the performance of the MLP AF classifiers trained on Fisher and retrained using alignments generated by AF-based recognizers. The initial set of classifiers, trained on Fisher phonetic alignments, were described in Section 4.2. Table 5.3 shows the accuracies of these initial AF MLPs on the 78 manually-transcribed SVitchboard utterances, scored against both the manual transcriptions and the forced phone alignments converted to features. The accuracies relative to the WS06 transcribers are very similar, and are lower than the accuracies relative to the forced alignments. This is to be expected, since the MLPs were trained on forced phonetic alignments. As an example of the types of confusions observed, Figure 5.8 displays the confusion matrices for **dg1** and **pl1**, relative to both the forced alignments and to Transcriber 1's labels.

Reference	**Accuracy (%)**					
	dg1	glo	nas	pl1	rou	vow
Forced align.	78/41	87/53	97/95	78/41	94/91	83/74
Transcriber 1	74/47	86/57	95/94	74/47	92/90	78/73
Transcriber 2	73/47	86/57	94/93	72/47	92/90	77/71

Table 5.3: Accuracies of the MLP AF classifiers evaluated against forced alignments and against the WS06 transcribers. The format of the table cells is "[MLP accuracy]/[chance accuracy]", where chance accuracy is defined as the accuracy that would be obtained by choosing the most common label value in the reference data.

We also evaluated the MLPs' performance after retraining from alignments generated using the hybrid and "monofeat" systems (Figure 5.9). For these results we excluded regions of silence from the calculation of frame accuracy. After retraining, all classifiers show a decrease in frame accuracy when scored against the phone alignments converted to features. However, when scoring against the manual feature transcriptions, the classifiers trained on hybrid alignments show an improvement for **dg1**, **pl1**, and **nas**; and the classifiers trained on monofeat alignments show improvement for **nas** and **vow** and a negligible decrease in accuracy for **glo** and **rou**.

5.3 Generation and analysis of forced AF alignments

It is also interesting to consider the accuracies of the forced AF alignments themselves, in order to study to what extent we are improving the quality of the training data when we re-align. By "forced AF alignments", we simply mean the Viterbi-decoded values of the AF variables for all time frames, when the word sequence is known. This is exactly analogous to forced phone alignment, but for multiple feature streams. In our case we use the Mississippi State word segmentations in producing the alignments, so we simply assume that the word variable is observed in every frame.

Figure 5.10 shows, for each of several sets of alignments, the proportion of time in agreement with one of the manual transcribers (the results are very similar when comparing to either transcriber). The alignments shown are as follows:

- The original SRI phone alignments converted to AF values. These were done without the benefit of the word segmentations, so they are not directly comparable to the AF alignments produced by our models. This is included in the figure as a reference.
- Alignments produced using the hybrid AF-based model with a monophone pronunciation model plus non-deterministic mapping to AFs. These are the alignments used for the embedded training of Section 4.3.3.
- Alignments produced using the 1-state and 3-state monofeat models of Sections 3.1 and 3.2, converted deterministically from the {L, T, G} feature set to the feature set used in the manual transcriptions.
- Alignments produced using a "hybrid monofeat" model. This is a new kind of model, not tested in recognition experiments, using the monofeat pronunciation model, a deterministic mapping from the pronunciation modeling AF set to the observation modeling AF set, and virtual evidence from the MLPs. This model, shown in Figure 5.11, is a hybrid model that also allows for articulatory asynchrony. This is the only model we know of that combines AF-based pronunciation and observation models.

There are several noteworthy aspects of these results. First, we note that the "hybrid monophone" alignments are less accurate, relative to human labels, than any of the other alignments including the original phone alignments. This should not be too surprising, since the non-deterministic phone-to-AF mapping in this model allows the AFs to change values in each frame independently of neighboring frames. Nevertheless, using the hybrid monophone alignments for embedded training of the MLPs improved the performance of the hybrid recognizer. This attests to the fact that the goal of embedded training is not to obtain better

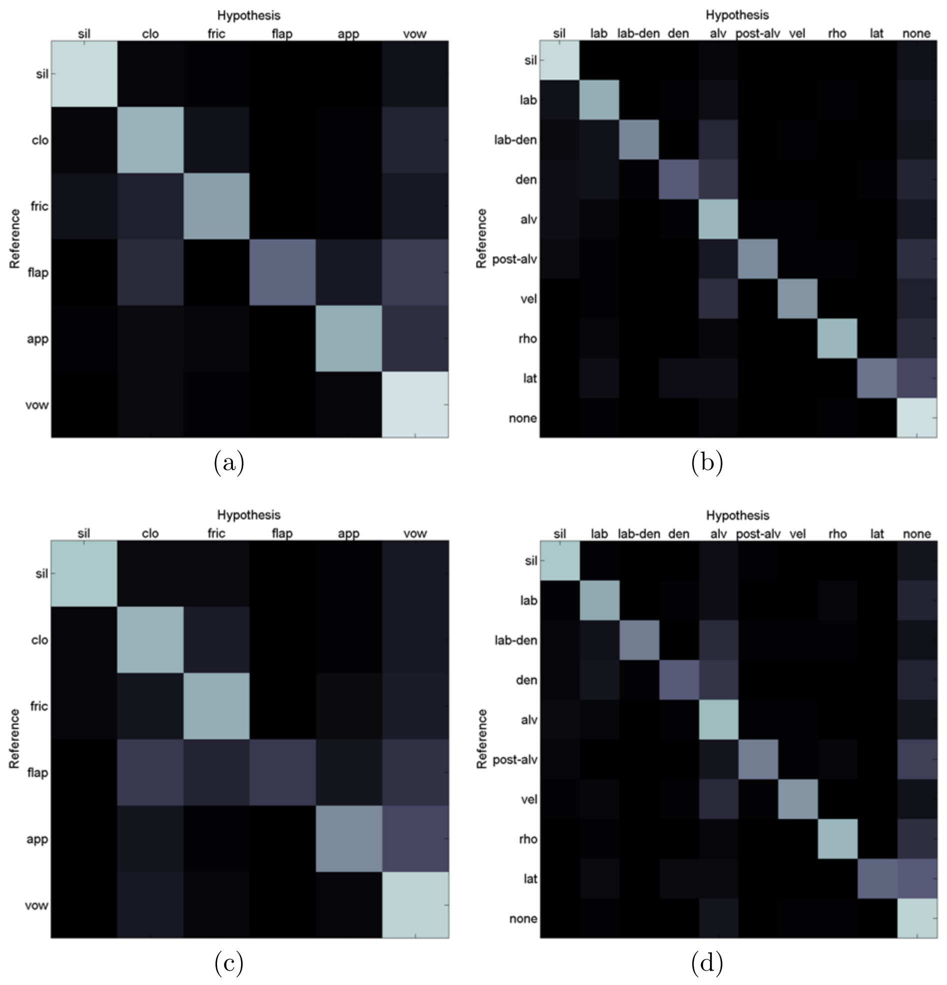


Figure 5.8: Confusion matrices of MLP classifier outputs for **dg1** (left) and **pl1** (right). In each figure, the MLP outputs are the “hypothesis”. The “reference” is either forced phone alignments converted to feature values (top) or Transcriber 1 (bottom).

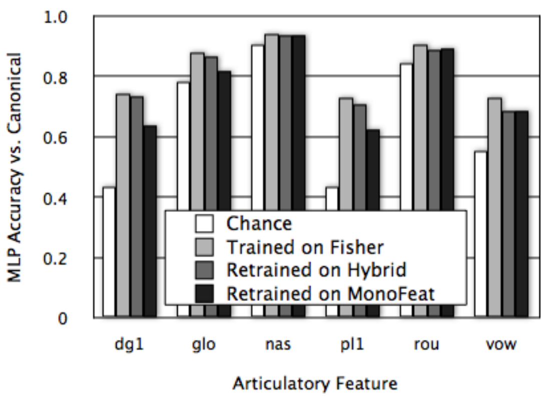
alignments but to improve the recognizer’s performance. The new alignments may be a better “fit” to the model despite being less “correct”.

Next we note that the 1-state monofeat recognizer, despite having far worse recognition performance than any of the other models, produces alignments that are almost as accurate as the most accurate alignments. The highest accuracies are usually produced by either the 3-state monofeat or the hybrid monofeat model, depending on the feature.

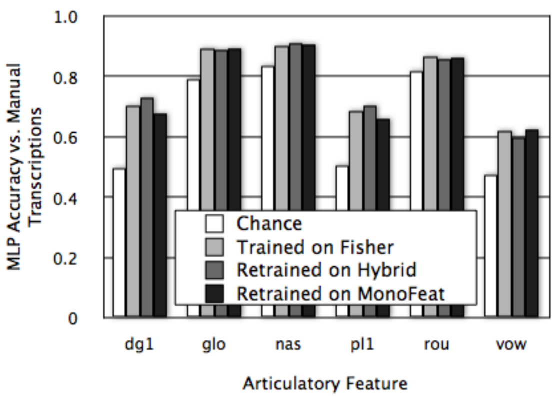
5.4 Discussion

In this chapter, we have, for the first time to our knowledge, tested AF classifiers and AF alignments against human labels. We have developed an approach for human AF labeling of speech that produces labels with high inter-transcriber agreement, and we have produced a small set of human AF transcriptions of SVitchboard and STP data. This set is not large enough to be used for training, but is large enough for testing classifiers and alignments.

We have observed that when our MLP AF classifiers are retrained, their outputs often move closer to the human labels while moving farther from the original phonetic alignments. This shows that the retraining is



(a)



(b)

Figure 5.9: MLP accuracies relative to phone alignments converted to canonical feature values (a) and relative to manual transcriptions (b) before and after retraining using alignments generated by the hybrid and monofeat systems.

doing what it should, and also demonstrates the usefulness of the human labels: It is simply not correct to measure AF classifier performance against phone alignments when those phone alignments do not allow for the phenomena of asynchrony and feature change that may be occurring in the data.

We have also observed, by testing forced AF alignments produced by different AF-based models, that it is not necessarily the best decoding models that produce the best alignments. While the same models can be used both for decoding and for producing alignments, the two tasks are different and may benefit from different models. It was encouraging to see the monofeat models, which have had a lackluster decoding performance thus far, produce some of the best AF alignments. In the long term, such alignments may be useful in their own right: If accurate AF alignments could be produced, they would have many applications such as to provide feedback in language learning scenarios, diagnosis of certain disorders, or analysis of speaker and context variations in production. When used to generate forced alignments, the AF models are less computationally intensive (since the words need not be inferred), making it feasible to use more complex and powerful models for alignment than for decoding.

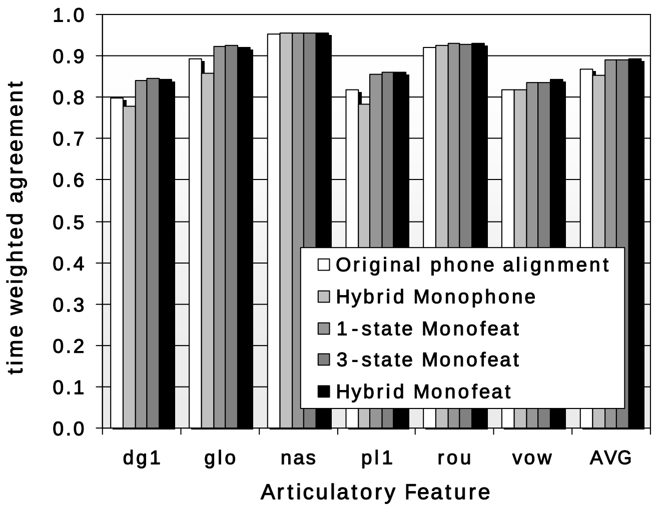


Figure 5.10: Accuracy of AF forced alignments generated by several models, measured as time in agreement with one of the manual transcribers. The final category, “AVG”, is the mean accuracy over all features.

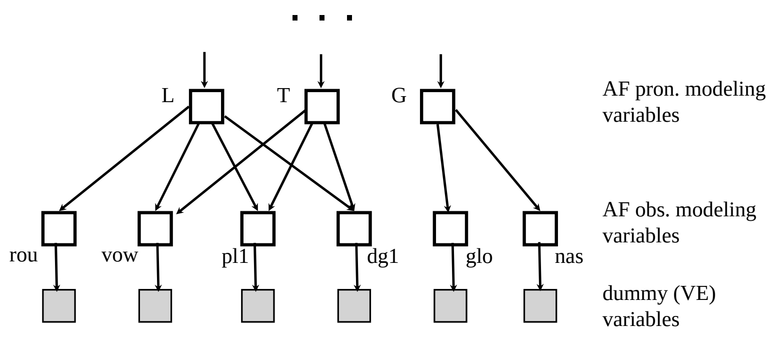


Figure 5.11: A “hybrid monofeat” model. This model is identical to a monofeat model, with the addition of the observation model AF variables (pl1, dg1, etc.) and the virtual evidence variables. Only one frame is shown, including only the AF variables.

Chapter 6

Conclusions

In this project, we have focused on several directions of research related to articulatory feature-based speech recognition. While most prior work has focused on using AF classifiers in combination with an otherwise phone-based recognizer (with some exceptions [47, 21, 73, 57, 54]), we have investigated the use of AFs in both the observation and pronunciation models. We have implemented all of our models using dynamic Bayesian networks, placing them in a unified framework and making them easily comparable and reconfigurable. We have also applied, for the first time to our knowledge, AF-based models to audio-visual continuous speech recognition. Finally, we have collected manual transcriptions of AFs and used them, for the first time, as reference transcripts in analysis of classifier performance and forced AF alignments. As a by-product of developing feature sets for labeling and recognition of conversational speech, we believe we now have more principled and complete feature sets for describing this type of speech than the ones we started with at the outset of this project. In the process, we have also developed tools (gmtkTie, GMTKtoWaveSurfer, site-independent parallel training and decoding scripts) that provide infrastructure for research in this area and with DBNs in general. Many materials related to this project, including the manual transcriptions, are available for download from `http://people.csail.mit.edu/klivescu/twiki/bin/view.cgi/WS06`.

In the area of pronunciation modeling with multistream AF-based models, we have studied a number of DBN models in experiments with SVitchboard data. In most of these experiments, we used models that allow for context-independent AF asynchrony within word boundaries. We have seen large improvements in recognition performance throughout the course of the workshop, but the AF-based models alone are still not outperforming phone-based models. One of the main areas for further work is the appropriate training of the rarer asynchronous states; possible ideas we have begun to explore are improved initializations and AF-based state tying. We have also begun to investigate more complex models, such as ones allowing for feature value substitutions and cross-word asynchrony. Such additions may in fact be necessary to realize the full benefit of AF-based models.

When applied to audio-visual recognition on digit strings from the CUAVE database, AF-based models match phoneme-viseme models. A future extension of the audio-visual work is to apply similar models to more complex tasks, where the effects of non-canonical articulations may be more pronounced.

In the area of observation modeling using AF classifiers, we have studied both hybrid and tandem models. The hybrid models are worse than the baseline phone models, and match the baseline with the addition of the PLP observations. Embedded training substantially improves recognition performance. The small number of parameters in hybrid models may make them most promising for cross-domain or cross-lingual ASR, where the use of a number of "universal" classifiers that apply cross-domain or cross-lingually may minimize the requirements for task-specific training data.

Similarly to hybrid models, tandem models use discriminative classifiers, but in contrast, they use Gaussian mixture-based models of the post-processed classifier outputs. The tandem AF-based models we developed significantly outperform monophone and triphone baselines, and match the performance of phone-based tandem models. Factoring the tandem observation models significantly improves performance over the unfactored models. Work is ongoing on additional factoring experiments, such as using a separate factor for each feature.

Finally, we have collected a set manual AF-level transcriptions for Switchboard utterances to analyze the performance of AF classifiers and alignments produced by different models. This enables us, for the first time, to measure AF classification performance against human-labeled references and to meaningfully measure differences in performance after retraining. We have found that the outputs of AF classifiers, when retrained using transcriptions generated by AF-based models, tend to move toward the manual labels and away from the original phone alignments. In examining AF alignments, we have seen that some of the best alignments are produced by the worst recognizers in terms of WER, and vice versa. This suggests that different models may be best suited to different tasks. Considering the potential usefulness of AF alignments for various applications, future work may include optimizing models specifically for the task of forced AF decoding.

In the longer term, we expect that the best gains will be realized by combining both AF-specific observation models and AF-based pronunciation models. We have in fact built such a combined model, and tested its AF alignment performance (Section 5.3), but have not yet used or optimized it for decoding. Ongoing work is focusing on various aspects of this effort, including adding context-dependency and flexibility to our pronunciation models, extending the audio-visual recognition work to more complex tasks, and porting the AF classifier-based observation models to other languages.

Appendix A

Phone-to-Articulatory Feature Mapping for Pronunciation Modeling Experiments

Table A.1 defines the feature set used in pronunciation modeling experiments. The table is reproduced from Section 2.1.1 but also provides a numerical index for each feature value, used in Table A.5 to specify the mapping from phone states to features used in all SVitchboard experiments. For most phones, the feature values are identical for all three states; for diphthongs, stops, and affricates, there are different feature values for the beginning and ending states. For some audio-visual experiments, the feature set was modified (see Section 3.4).

Feature	Description	# values	Values
L	Combination of LIP-LOC and LIP-OPEN	6	0=P-N, 1=P-W, 2=L-CL, 3=L-CR, 4=L-W, 5=D-CR
T	Combination of TT-LOC, TT-OPEN, TB-LOC, and TB-OPEN	19	0=D-CR-U-M, 1=A-CL-U-N, 2=A-CL-U-M, 3=A-CR-U-M, 4=A-N-U-M, 5=A-MN-PA-N, 6=A-MN-PA-MN, 7=A-M-PA-M, 8=A-M-U-M, 9=A-W-V-M, 10=P-CR-PA-MN, 11=P-M-U-MN, 12=P-W-V-CL, 13=P-W-V-CR, 14=P-W-V-N, 15=P-W-U-N, 16=P-W-U-MN, 17=P-W-PH-MN, 18=R-N-U-M
G	Combination of VEL and GLOT	3	0=C-VO, 1=C-VL, 2=O-VO

Table A.1: Definition of the articulatory feature set used for pronunciation modeling. See Section 2.1.1 for additional details.

Phone state	Index	L	T	G
aa_0	0	4	17	0
aa_1	0	4	17	0
aa_2	0	4	17	0
ae_0	1	4	9	0
ae_1	1	4	9	0
ae_2	1	4	9	0
ah_0	2	4	8	0
ah_1	2	4	8	0
ah_2	2	4	8	0
ao_0	3	1	17	0
ao_1	3	1	17	0
ao_2	3	1	17	0
aw1_0	4	4	9	0
aw1_1	4	4	9	0
aw2_0	5	0	16	0
ax_0	6	4	8	0
ax_1	6	4	8	0
ax_2	6	4	8	0
axr_0	7	4	18	0
axr_1	7	4	18	0
axr_2	7	4	18	0
ay1_0	8	4	17	0
ay1_1	8	4	17	0
ay2_0	9	4	6	0
b_0	10	3	8	0
bcl_0	11	2	8	0
bcl_1	11	2	8	0
ch_0	12	4	10	1
ch_1	12	4	10	1
ch_2	12	4	10	1
d_0	13	4	3	0
dcl_0	14	4	2	0
dcl_1	14	4	2	0
dh_0	15	4	0	0
dh_1	15	4	0	0
dh_2	15	4	0	0
dx_0	16	4	4	0
dx_1	16	4	4	0
dx_2	16	4	4	0
eh_0	17	4	7	0
eh_1	17	4	7	0
eh_2	17	4	7	0
el_0	18	4	1	0
el_1	18	4	1	0
el_2	18	4	1	0

Table A.2: Mapping from phone states to feature values used in SVitchboard pronunciation modeling experiments. Continued on the following pages.

Phone state	Index	L	T	G
em_0	19	2	8	2
em_1	19	2	8	2
em_2	19	2	8	2
en_0	20	4	2	2
en_1	20	4	2	2
en_2	20	4	2	2
er_0	21	4	18	0
er_1	21	4	18	0
er_2	21	4	18	0
ey1_0	22	4	7	0
ey1_1	22	4	7	0
ey2_0	23	4	6	0
f_0	24	5	8	1
f_1	24	5	8	1
f_2	24	5	8	1
g_0	25	4	13	0
gcl_0	26	4	12	0
gcl_1	26	4	12	0
hh_0	27	4	8	1
hh_1	27	4	8	1
hh_2	27	4	8	1
ih_0	28	4	6	0
ih_1	28	4	6	0
ih_2	28	4	6	0
iy_0	29	4	5	0
iy_1	29	4	5	0
iy_2	29	4	5	0
jh_0	30	4	10	0
jh_1	30	4	10	0
jh_2	30	4	10	0
k_0	31	4	13	1
kcl_0	32	4	12	1
kcl_1	32	4	12	1
l_0	33	4	1	0
l_1	33	4	1	0
l_2	33	4	1	0
m_0	34	2	8	2
m_1	34	2	8	2
m_2	34	2	8	2

Table A.3: Mapping from phone states to feature values used in SVitchboard pronunciation modeling experiments. Continued from the previous page.

Phone state	Index	L	T	G
n_0	35	4	2	2
n_1	35	4	2	2
n_2	35	4	2	2
ng_0	36	4	12	2
ng_1	36	4	12	2
ng_2	36	4	12	2
ow1_0	37	1	16	0
ow1_1	37	1	16	0
ow2_0	38	0	15	0
oy1_0	39	1	16	0
oy1_1	39	1	16	0
oy2_0	40	4	6	0
p_0	41	3	8	1
pcl_0	42	2	8	1
pcl_1	42	2	8	1
r_0	43	4	18	0
r_1	43	4	18	0
r_2	43	4	18	0
s_0	44	4	3	1
s_1	44	4	3	1
s_2	44	4	3	1
sh_0	45	4	10	1
sh_1	45	4	10	1
sh_2	45	4	10	1
sil_0	57	2	2	1
sil_1	57	2	2	1
sil_2	57	2	2	1
t_0	46	4	3	1
tcl_0	47	4	2	1
tcl_1	47	4	2	1
th_0	48	4	0	1
th_1	48	4	0	1
th_2	48	4	0	1
uh_0	49	1	11	0
uh_1	49	1	11	0
uh_2	49	1	11	0
uw_0	50	0	14	0
uw_1	50	0	14	0
uw_2	50	0	14	0

Table A.4: Mapping from phone states to feature values used in SVitchboard pronunciation modeling experiments. Continued from the previous page.

Phone state	Index	L	T	G
v_0	51	5	8	0
v_1	51	5	8	0
v_2	51	5	8	0
w_0	52	0	15	0
w_1	52	0	15	0
w_2	52	0	15	0
y_0	53	4	5	0
y_1	53	4	5	0
y_2	53	4	5	0
z_0	54	4	3	0
z_1	54	4	3	0
z_2	54	4	3	0
zh_0	55	4	10	0
zh_1	55	4	10	0
zh_2	55	4	10	0

Table A.5: Mapping from phone states to feature values used in SVitchboard pronunciation modeling experiments. Continued from the previous page.

Appendix B

Manual transcription guidelines

Below are the instructions used in the manual articulatory feature labeling. Further details on the data collection are provided in Section 5.1.

Checklist

- Before starting an utterance:
  - Point your web browser to the phone-to-feature mappings and the instructions on this page smile
  - Open the .wav file with sampling rate 8000 and offset 1024 bytes. (To avoid doing this separately for each file, you can try checking the "use these settings for all .wav files" box when opening the first file.)
  - Write down the time.
- To keep in mind during transcription:
  - The detailed guidelines below.
  - The boundaries in the initial .wd transcription were generated automatically and have many errors. You will probably need to modify some/many of them.
  - The provided initial/final silence boundaries in all tiers were generated automatically from the .wd files; these are also likely to be wrong.
  - The initial .phn transcription is an alignment with the dictionary pronunciations. It is often wrong when words are reduced. Change it with abandon.
  - The final transcription should give enough information so that the speaker, by looking at the transcription, could recreate the acoustics exactly. More on this in the detailed guidelines below.
  - Go for accuracy over speed. But if you are very unsure about a segment, use "?" or multiple labels (e.g. "FRIC/APP").
  - Don't worry about exact boundaries up to +/- 20ms.
  - Use the .cm tier to mark anything problematic or that should be discussed.
  - Use the drop-down menus to label feature tiers, rather than typing in.
  - It's OK to use some deductive reasoning. E.g. if the intended sound is [b] but the actual segment is a fricative, it is probably a LAB, FRIC and not a [v] (L-D, FRIC).
- When finishing an utterance:
  - Write down the time.
  - Save all transcription panes.

- Sanity checks:
  - Re-skim the guidelines below to make sure the transcription follows them.
  - Listen to each segment (using right-click on transcription → "Play label") to make sure you believe the transcription
- Stretch, get a drink...

- When doing the 2nd pass:
  - Fix mistakes, not disagreements.
  - Once the 2nd pass is done, you should be able to "defend" each difference between the transcriptions, i.e. "I chose this over that because...".

Detailed guidelines

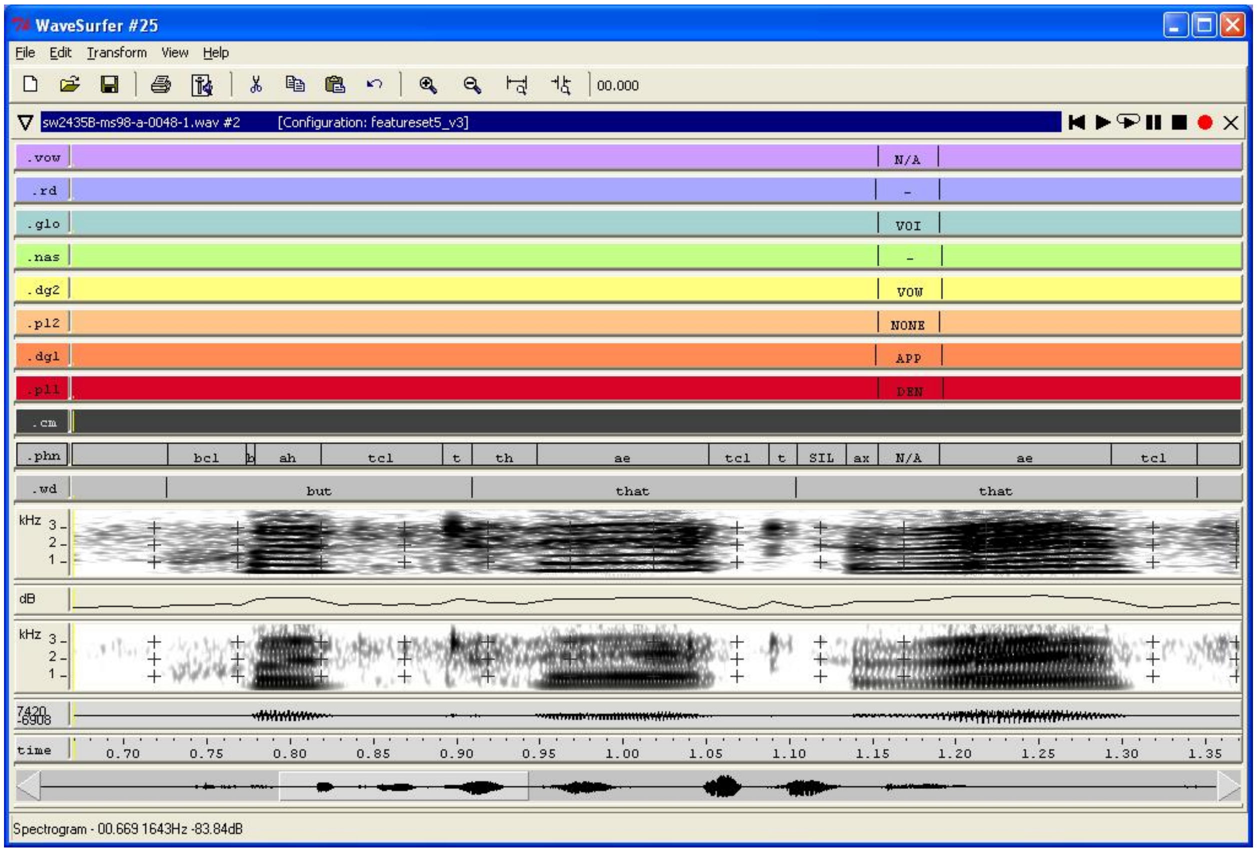
In no particular order...

- Phone/feature hybrid transcriptions:
  - Use a phone label when a segment's features match one of the phone labels. If they don't, mark "N/A" in the .phn tier and use the label tiers instead.
  - If there are multiple "N/A" segments in a row (i.e. segments with different feature values with no matching phone labels), leave them as multiple segments rather than making one big "N/A" segment. (No good reason for this; just a convention.)
  - If a phone has an unspecified feature value ([hh], [q], [r], [er], [axr], [sh], [zh], [ch], [jh]), it must be transcribed with both a phone label and feature values (for all tiers, not just the unspecified ones). E.g. an [hh] must be labeled with features since its vowel value is unspecified; it should be easy to tell what vowel shape the [hh] is in based on formants/listening. Similarly, when [q] is realized as IRR, the vowel shape is usually easy to tell and should be labeled in the .vow tier (if it's not easy to tell, label it '?'). For some phones with unspecified feature values, it may be hard to tell what the actual value is; e.g. an [r] may be rounded or unrounded; label it '?' if you can't tell.
- The "recreation" rule:
  - The final transcription should give enough information so that the speaker, by looking at the transcription, could recreate the acoustics exactly (assuming he/she could actually read the transcription).
  - For example, if a word is very reduced but with a hint of the original gestures, that should be indicated somehow in the transcription. E.g. if the word is "probably" and is produced like "pry" but with a hint of labial/lateral gestures in the middle, don't transcribe it as /pcl p r ay1 ay2/. Use place=LAB or LAT and degree=APP to indicate these hints of gestures.
- [ah] vs. [ax], [ih] vs. [ix], [er] vs. [axr]: Use a schwa if the segment is unstressed and 50ms or shorter.
- APP degree:
  - Used for both glides ([y], [w]) and other sounds realized as approximants.
  - If there is any gesture towards an intended consonant, even if small, use APP degree. E.g. if "probably" is produced almost like [p r ay] but with some evidence of lip narrowing in the middle of the [ay]-like region, mark that as APP, LAB.
- Two stop closures in a row: If you can't tell when the place of closure has changed (e.g. "woul*d g*o"), just mark the boundary in the middle.

- Baseline phonetic transcription: The initial transcription is an alignment (currently, done by Karen) of the utterance to its baseform (dictionary) pronunciation. In other words, "probably" will always get the initial phonetic transcription [p r aa bcl b ax bcl b l iy] even if it was pronounced [p r ay]. This transcription is a very very rough guideline; ignore it or change it as much as necessary to match what was actually spoken.
- "GLO" place: Used only for glottal stops.
- VOI vs. IRR in the .glo tier: If there are regular pitch periods, even with very low pitch, label them as VOI. Use IRR only when the pitch periods are not at regular intervals.
- Boundaries in diphthongs: Where does the boundary between 1 and 2 go?
  - [aw1] should look/sound more like an [ae] (or [aa] in some dialects), [aw2] like an [uh] or [w]
  - [ay1] ↔ [aa], [ay2] ↔ [ih] or [y]
  - [ey1] ↔ [eh], [ey2] ↔ [ih] or [y]
  - [ow1] doesn't have a non-diphthong correlate, [ow2] ↔ [uh] or [w]
  - [oy1] ↔ [ao] or [ow1], [oy2] ↔ [ih] or [y]
- Voiceless vowels: mark them as "VL", not "ASP", in the glottal tier (to differentiate from [hh])
- The vowel rule:
  - If the .pl/.dg tiers are both "NONE/VOW", then there should be a vowel label in the .vow tier; otherwise, the .vow tier should be "N/A".
  - The only exception is for rhoticized vowels ([er], [axr]) and syllabics ([el], [em], [en]); these get a vowel label in the .vow tier and a constriction in the .pl1/.dg1 tier ("RHO/APP" for [axr], [er]; "LAT/CLO", "LAB/CLO", or "ALV/CLO" for [el], [em], [en]).
- Laterals: Use "LAT" place for both light and dark [l]s. For an [l] with incomplete tongue tip closure, use "APP" in the degree tier. Example: the [l]s at the end of the words "all", "feel" would usually be marked as a "LAT/APP" segment followed by a "LAT/CLO" segment.
- Stops:
  - For unaspirated (voiced or voiceless) stops, the .dg is CLO during the closure, then FRIC during the frication.
  - For voiceless stops at the beginning of a stressed syllable, there may be aspiration. If there is a clear distinction between a frication portion and aspiration portion, use CLO for closure, FRIC for frication, APP during the aspiration. The aspiration should also be indicated as ASP in the .glo tier.
  - When doing a phonetic labeling, the burst is not separated into frication and aspiration; however, if there are feature changes during the aspiration (e.g. if the lateral in the word "place" occurs during the aspiration), then label the aspiration portion with feature labels.
  - If there is an utterance-initial or -final stop closure, it may not be possible to tell where the boundary between closure and silence is. In that case, mark a 100ms-long closure.
- Transitional periods between steady states: Don't label them as separate segments if they are natural/necessary transitional periods, e.g. the formant transitions between vowels and consonants. If there is an "extra" transitional sound beyond what is necessary, like "feel" → [f iy ax l], label it as such.
- Voicing onset/offset: When in doubt, open a waveform blow-up; the onset/offset of voicing is the point at which periodicity starts/stops.

- Diphthongs realized as monophthongs: Label them as the monophthong, not as 1, when possible. For example, an underlying [aw] that's produced as an [ae] should be marked [ae], not [aw1]. Similarly for [ey]. For [ow], there's no monophthong label corresponding to the first part, so mark it [ow1]. For [oy], the initial part of it may sound like an [ao], in which case label it as such; or it may sound more like the sound at the beginning of [ow] or "or", in which case label it [oy1].

- Here's a tricky utterance we talked about at the May 10 meeting, and the transcription I think we agreed on (at least I think we agreed on the [dh] portion, which was the tricky part).


WaveSurfer #25
File Edit Transform View Help
00.000
sw2435B-ms98-a-0048-1.wav #2 [Configuration: featureset5_v3]
.vow N/A
.rd -
.glo VOI
.nas -
.dg2 VOW
.pl2 NONE
.dg1 APP
.pl1 DEN
.cm
.phn bcl b ah tcl t th ae tcl t SIL ax N/A ae tcl
.wd but that that
kHz 3 2 1
dB
kHz 3 2 1
7420 -6908
time 0.70 0.75 0.80 0.85 0.90 0.95 1.00 1.05 1.10 1.15 1.20 1.25 1.30 1.35
Spectrogram - 00.669 1643Hz -83.84dB

Bibliography

[1] A. Adjoudani and C. Benoit. On the integration of auditory and visual parameters in an HMM-based ASR. In D. G. Stork and M. E. Hennecke, editors, *Speechreading by Humans and Machines: Models, Systems, and Applications*, pages 461–471. Springer, New York, 1996.

[2] R. W. Albright. The International Phonetic Alphabet: Its background and development. *International Journal of American Linguistics*, 24(1, part 3), 1958.

[3] C. Bartels, K. Duh, J. Bilmes, K. Kirchhoff, and S. King. Genetic triangulation of graphical models for speech and language processing. In *Proc. Eurospeech*, 2005.

[4] Y. Bengio. *Neural Networks for Speech and Sequence Recognition.* International Thomson Computer Press, London, 1995.

[5] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe. Global optimization of a neural network - hidden Markov model hybrid. *IEEE Transactions on Neural Networks*, 3(2):252–259, 1992.

[6] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe. Phonetically motivated acoustic parameters for continuous speech recognition using artificial neural networks. *Speech Communication*, 11(2-3):261–271, 1992.

[7] J. Bilmes. Dynamic Bayesian multinets. In *Proc. Uncertainty in Artificial Intelligence (UAI)*, Stanford, CA, July 2000.

[8] J. Bilmes. Buried Markov models: A graphical-modeling approach to automatic speech recognition. *Computer Speech and Language*, 17(2–3), 2003.

[9] J. Bilmes and C. Bartels. Graphical model architectures for speech recognition. *IEEE Signal Processing Magazine*, 22(5):89–100, 2005.

[10] J. Bilmes and G. Zweig. The Graphical Models Toolkit: An open source software system for speech and time-series processing. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Orlando, FL, May 2002.

[11] H. Bourlard and S. Dupont. A new ASR approach based on independent processing and recombination of partial frequency bands. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, Philadelphia, October 1996.

[12] C. P. Browman and L. Goldstein. Articulatory phonology: An overview. *Phonetica*, 49:155–180, 1992.

[13] G. Calvert, M. Brammer, E. Bullmore, R. Campbell, S. Iversen, and A. David. Response amplification in sensory-specific cortices during crossmodal binding. *Neuroreport*, 10:2619–2623, 1999.

[14] G. Calvert, E. Bullmore, M. Brammer, R. Campbell, S. Williams, P. McGuire, P. Voodruff, S. Iversen, and A. David. Activation of auditory cortex during silent lipreading. *Science*, 276:593–596, 1997.

[15] O. Cetin, A. Kantor, S. King, C. Bartels, M. Magimai-Doss, J. Frankel, and K. Livescu. An articulatory feature-based tandem approach and factored observation modeling. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[16] S. Chang, S. Greenberg, and M. Wester. An elitist approach to articulatory-acoustic feature classification. In *Proc. Eurospeech*, 2001.

[17] S. Chu and T. S. Huang. Bimodal speech recognition using coupled hidden Markov models. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, Beijing, 2000.

[18] S. M. Chu and T. S. Huang. Multi-modal sensory fusion with application to audio-visual speech recognition. In *Proc. Eurospeech*, 2001.

[19] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46, 1960.

[20] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[21] L. Deng, G. Ramsay, and D. Sun. Production models as a structural basis for automatic speech recognition. *Speech Communication*, 33:93–111, 1997.

[22] D. P. W. Ellis, R. Singh, and S. Sivadas. Tandem acoustic modeling in large-vocabulary recognition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2001.

[23] J. Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 1997.

[24] C. G. Fisher. Confusions among visually perceived consonants. *J. Speech and Hearing Research*, 11(4):796–804, 1968.

[25] C. A. Fowler and D. J. Dekle. Listening with eye and hand: Crossmodal contributions to speech perception. *Journal of Experimental Psychology: Human Perception and Performance*, 17:816–828, 1991.

[26] J. Frankel, M. Magimai-Doss, S. King, K. Livescu, and O. Cetin. Articulatory feature classifiers trained on 2000 hours of telephone speech. In *Proc. Interspeech*, 2007.

[27] N. Ganapathiraju, A. Deshmukh, A. Gleeson, J. Hamaker, and J. Picone. Resegmentation of SWITCHBOARD. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, 1998.

[28] Z. Gharamani and M. I. Jordan. Factorial hidden Markov models. *Machine Learning*, 29:245–273, 1997.

[29] L. Gillick and S. J. Cox. Some statistical issues in the comparison of speech recognition algorithms. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1989.

[30] GMTK. http://ssli.ee.washington.edu/$\tilde{\ }$bilmes/gmtk/.

[31] J. J. Godfrey, E. C. Holliman, and J. McDaniel. SWITCHBOARD: Telephone speech corpus for research and development. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, San Francisco, March 1992.

[32] J. A. Goldsmith. *Autosegmental and metrical phonology.* Basil Blackwell, Oxford, UK, 1990.

[33] J. N. Gowdy, A. Subramanya, C. Bartels, and J. Bilmes. DBN-based multi-stream models for audio-visual speech recognition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2004.

[34] K. W. Grant and P. F. Seitz. The recognition of isolated words and words in sentences: Individual variability in the use of sentence context. *Journal of the Acoustical Society of America*, 107:1000–1011, 2000.

[35] K. W. Grant, B. E. Walden, and P. F. Seitz. Auditory-visual speech recognition by hearing-impaired subjects: Consonant recognition, sentence recognition, and auditory-visual integration. *Journal of the Acoustical Society of America*, 103:2677–2690, 1998.

[36] G. Gravier, G. Potamianos, and C. Neti. Asynchrony modeling for audio-visual speech recognition. In *Proc. HLT*, San Diego, CA, March 2002.

[37] S. Greenberg, J. Hollenback, and D. Ellis. Insights into spoken language gleaned from phonetic transcription of the Switchboard corpus. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, Philadelphia, October 1996.

[38] M. Hasegawa-Johnson, J. Baker, S. Borys, K. Chen, E. Coogan, S. Greenberg, A. Juneja, K. Kirchhoff, K. Livescu, K. Sonmez, S. Mohan, J. Muller, and T. Wang. Landmark-based speech recognition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005.

[39] M. Hasegawa-Johnson, J. Baker, S. Greenberg, K. Kirchhoff, J. Muller, K. Sonmez, S. Borys, K. Chen, A. Juneja, K. Livescu, S. Mohan, E. Coogan, and T. Wang. Landmark-based speech recognition: Final report. Technical report, Johns Hopkins University Center for Language and Speech Processing, 2004.

[40] M. Hasegawa-Johnson, K. Livescu, P. Lal, and K. Saenko. Audiovisual speech recognition with articulator positions as hidden variables. In *Proc. International Congress of Phonetic Sciences (ICPhS)*, 2007.

[41] H. Hermansky, D. P. W. Ellis, and S. Sharma. Tandem connectionist feature extraction for conventional HMM systems. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2000.

[42] HTK. http://htk.eng.cam.ac.uk/.

[43] P.L. Jackson. The theoretical minimal unit for visual speech perception: Visemes and coarticulation. In *New Reflections on Speechreading*, volume 90, pages 99–115. Alexander Graham Bell, Washington, DC, 1988.

[44] A. Juneja and C. Espy-Wilson. Significance of invariant acoustic cues in a probabilistic framework for landmark-based speech recognition. In *From Sound to Sense: Fifty+ Years of Discoveries in Speech Communication*, MIT, Cambridge, Massachusetts, 2004.

[45] S. King, J. Bilmes, and C. Bartels. SVitchboard: Small-vocabulary tasks from Switchboard. In *Proc. Interspeech*, 2005.

[46] S. King and P. Taylor. Detection of phonological features in continuous speech using neural networks. *Computer Speech and Language*, 14(4):333–354, 2000.

[47] K. Kirchhoff. Syllable-level desynchronisation of phonetic features for speech recognition. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, Philadelphia, PA, October 1996.

[48] K. Kirchhoff. *Robust Speech Recognition Using Articulatory Information.* PhD dissertation, University of Bielefeld, Germany, 1999.

[49] K. Kirchhoff, G. A. Fink, and G. Sagerer. Combining acoustic and articulatory feature information for robust speech recognition. *Speech Communication*, 37:303–319, 2002.

[50] J. R. Landis and G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174, 1977.

[51] B. Lee, M. Hasegawa-Johnson, C. Goudeseune, S. Kamdar, S. Borys, M. Liu, and T. Huang. AVICAR: Audio-visual speech corpus in a car environment. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, 2004.

[52] X. Lei, M.-Y. Hwang, and M. Ostendorf. Incorporating tone-related MLP posteriors in the feature representation for Mandarin ASR. In *Proc. Interspeech*, 2005.

[53] T. W. Lewis. *Noise-Robust Audio-Visual Phoneme Recognition.* PhD dissertation, Flinders University, Adelaide, South Australia, 2005.

[54] K. Livescu. *Feature-based Pronunciation Modeling for Automatic Speech Recognition.* PhD dissertation, MIT, Cambridge, MA, 2005.

[55] K. Livescu, A. Bezman, N. Borges, L. Yung, O. Cetin, J. Frankel, S. King, M. Magimai-Doss, X. Chi, and L. Lavoie. Manual transcription of conversational speech at the articulatory feature level. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[56] K. Livescu, O. Cetin, M. Hasegawa-Johnson, S. King, C. Bartels, N. Borges, A. Kantor, P. Lal, L. Yung, A. Bezman, S. Dawson-Haggerty, B. Woods, J. Frankel, M. Magimai-Doss, and K. Saenko. Articulatory feature-based methods for acoustic and audio-visual speech recognition: Summary from the 2006 JHU Summer Workshop. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[57] K. Livescu and J. R. Glass. Feature-based pronunciation modeling for automatic speech recognition. In *Proc. HLT/NAACL*, Boston, May 2004.

[58] K. Livescu and J. R. Glass. Feature-based pronunciation modeling with trainable asynchrony probabilities. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, Jeju, South Korea, October 2004.

[59] D. Massaro. *Speech Perception by Ear and Eye: A Paradigm for Psychological Inquiry.* Erlbaum, Hillsdale, NJ, 1987.

[60] H. McGurk and J. MacDonald. Hearing lips and seeing voices. *Nature*, 264:746–748, 1976.

[61] F. Metze and A. Waibel. A flexible stream architecture for ASR using articulatory features. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, 2002.

[62] N. Morgan and H. Bourlard. Continuous speech recognition: An introduction to the hybrid HMM/connectionist approach. *IEEE Signal Processing Magazine*, 12(3):25–42, 1995.

[63] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning.* PhD dissertation, U. C. Berkeley, Berkeley, CA, 2002.

[64] A. V. Nefian, L. Liang, X. Pi, L. Xiaoxiang, C. Mao, and K. Murphy. A coupled HMM for audio-visual speech recognition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Orlando, Florida, May 2002.

[65] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7(4):308–313, 1965.

[66] C. Neti, G. Potamianos, J. Luettin, Iain Matthews, Hervé Glotin, Dimitra Vergyri, June Sison, Azad Mashari, and Jie Zhou. Audio-visual speech recognition: Final report. Technical report, Johns Hopkins University Center for Language and Speech Processing, 2000.

[67] P. Niyogi, E. Petajan, and J. Zhong. Feature based representation for audio-visual speech recognition. In *Proc. Audio Visual Speech Conference*, Santa Cruz, CA, 2005.

[68] H. J. Nock and S. J. Young. Modelling asynchrony in automatic speech recognition using loosely-coupled HMMs. *Cognitive Science*, 26(3):283–301, May–June 2002.

[69] J. Odell. *The Use Of Context In Large Vocabulary Speech Recognition.* PhD thesis, University of Cambridge, March 1995.

[70] M. Ostendorf. Incorporating linguistic theories of phonological variation into speech recognition models. *Philosophical Transactions of the Royal Society*, 358(1769):1325–1338, 2000.

[71] E. K. Patterson, S. Gurbuz, Z. Tufecki, and J. N. Gowdy. CUAVE: A new audio-visual database for multimodal human-computer interface research. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002.

[72] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, San Mateo, CA, 1988.

[73] M. Richardson, J. Bilmes, and C. Diorio. Hidden-articulator Markov models for speech recognition. *Speech Communication*, 41(2), 2003.

[74] R. C. Rose, J. Schroeter, and M. M. Sondhi. An investigation of the potential role of speech production models in automatic speech recognition. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, 1994.

[75] K. Saenko and K. Livescu. An asynchronous DBN for audio-visual speech recognition. In *Proc. IEEE Workshop on Spoken Language Technology (SLT)*, Palm Beach, Aruba, 2006.

[76] K. Saenko, K. Livescu, J. Glass, and T. Darrell. Production domain modeling of pronunciation for visual speech recognition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005.

[77] K. Saenko, K. Livescu, M. Siracusa, K. Wilson, J. Glass, and T. Darrell. Visual speech recognition with loosely synchronized feature streams. In *Proc. International Conference on Computer Vision (ICCV)*, Beijing, China, October 2005.

[78] M. Sams, R. Aulanko, H. Hamalainen, O. Lounasmaa, S. Lu, and J. Simola. Seeing speech: Visual information from lip movements modifies activity in the human auditory cortex. *Neuroscience Letters*, 127:141–145, 1991.

[79] H. Soltau, F. Metze, and A. Waibel. Compensating for hyperarticulation by modeling articulatory properties. In *Proc. International Conference on Spoken Language Processing (ICSLP)*, 2002.

[80] T. A. Stephenson, M. Magimai-Doss, and H. Bourlard. Speech recognition with auxiliary information. *IEEE Transactions on Speech and Audio Processing*, 12(3):189–203, 2004.

[81] A. Stolcke, F. Grĕzl, M.-Y. Hwang, X. Lai, M. Nelson, and D. Vergyri. Cross-domain and cross-language portability of acoustic features estimated by multilayer perceptrons. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2006.

[82] S. Stueker, F. Metze, T. Schultz, and A. Waibel. Integrating multilingual articulatory features into speech recognition. In *Proc. Eurospeech*, 2003.

[83] W. Sumby and I. Pollack. Visual contribution to speech intelligibility in noise. *Journal of the Acoustical Society of America*, 26:212–215, 1954.

[84] WaveSurfer. http://www.speech.kth.se/wavesurfer/.

[85] M. Wester, J. Frankel, and S. King. Asynchronous articulatory feature recognition using dynamic Bayesian networks. In *Proc. IEICI Beyond HMM Workshop*, Kyoto, Japan, December 2004.

[86] M. F. Woodward and C. G. Barber. Phoneme perception in lipreading. *J. Speech and Hearing Research*, 3(3):212–222, 1960.

[87] A. A. Wrench and W. J. Hardcastle. A multichannel articulatory speech database and its application for automatic speech recognition. In *Proc. 5th Seminar on Speech Production: Models and Data*, 2000.

[88] H. Yehia, P. Rubin, and E. Vatikiotis-Bateson. Quantitative association of vocal-tract and facial behavior. *Speech Communication*, 26:23–43, 1998.

[89] Y. Zhang, Q. Diao, S. Huang, W. Hu, C. Bartels, and J. Bilmes. DBN-based multi-stream models for speech. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2003.

[90] J. Zheng, Ö. Çetin, M.-Y. Hwang, X. Lei, A. Stolcke, and N. Morgan. Combining discriminative feature, transform, and model training for large vocabulary speech recognition. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2007.

[91] Q. Zhu, A. Stolcke, B. Y. Chen, and N. Morgan. Incorporating tandem/HATs MLP features into SRI's conversational speech recognition system. In *Proc. EARS RT-04F Workshop*, 2004.

[92] G. Zweig. *Speech recognition using dynamic Bayesian networks.* PhD dissertation, U. C. Berkeley, Berkeley, CA, 1998.