

Automatically Learning Speaker-independent Acoustic Subword Units

Balakrishnan Varadarajan and Sanjeev Khudanpur

Center for Speech and Language Processing, Johns Hopkins University, Baltimore, MD 21218

bvarada2@jhu.edu and khudanpur@jhu.edu

Abstract

We investigate methods for unsupervised learning of sub-word acoustic units of a language directly from speech. We demonstrate that states of a hidden Markov model “grown” using a novel modification of the maximum likelihood successive state splitting algorithm correspond very well with the phones of the language. In particular, the correspondence between the Viterbi state sequence for unseen speech from the training speaker and the phone transcription of the speech is over 85%, and generalizes to a large extent ($\sim 63\%$) to speech from a different speaker. Furthermore, we are able to bridge more than half the gap between the speaker-dependent and cross-speaker correspondence of the automatically learned units to phones ($\sim 75\%$ accuracy) by unsupervised adaptation via MLLR.

1. Introduction

Modern automatic speech recognition (ASR) systems rely on acoustic-phonetic models estimated from transcribed speech. However the phone vocabulary and phonetic pronunciations of words are hand-crafted from linguistic knowledge. This poses a challenge in extending current ASR methods to languages and dialects for which we do not have a comprehensive pronunciation dictionary or an adequate phone-set. We are also interested in extending methods from ASR to the recognition of signals other than speech, such as manipulative gestures in endoscopic surgery, where such “units” remain to be discovered. For recognition of complex gesture sequences in endoscopic surgery, it is necessary to derive reusable sub-gestural units, so that appropriately parsimonious gesture-models may be derived. With this alternate application in mind, we are investigating methods for automatically learning the sub-word units (i.e. the phone vocabulary) of a language directly from speech.

Svendsen and Soong [1] investigated dynamic programming approaches to segmenting speech, while Paliwal and Kulkarni [2] and researchers at IBM [3] used vector quantization of individual frames to label the speech, designating contiguous regions with identical labels as segments. Shikari and Honda [4] proposed a variable length segment quantizer that iteratively adjusts the segment boundaries and updates the codebook. After some early attempts of this kind, the problem of automatically deriving subword units was largely abandoned two decades ago in favor of manually specified phone vocabularies.

Attention has been given more recently to learning the allophonic variations of individual phonemes. Takami and Sagayama [5] proposed a successive state splitting algorithm to discover allophonic variations. Fukada and Bacchiani [6] proposed a new method to generate word models from acoustically derived segmental units based on combining statistically similar segments. Singer and Ostendorf [7] proposed a modified version of the state splitting algorithm and called it maximum likelihood successive state splitting (ML-SSS). We made some

further modifications to the ML-SSS algorithm to enhance its speed and to enable “lookahead” for searching a larger space of models. In this paper, we use this modified ML-SSS algorithm as the basis for automatically learning subword units of speech.

We begin, in Section 2, by reviewing our modified ML-SSS algorithm to automatically derive speech units from *untranscribed* speech. More formally we learn a finite state transducer (F_1) that maps continuous speech to discrete but as-yet-abstract labels. The labels are expected to model different acoustic-phonetic units of speech, possibly allophones.

We next investigate ways, in Section 3, to evaluate whether the labels learned by our algorithm are linguistically meaningful. To this end, we develop methods to learn a second finite state transducer (F_2) to automatically map (abstract) label sequences generated by F_1 to phone sequences, and measure the Levenshtein (edit) distance to a manual phonetic transcription of the speech. We note that while learning F_1 does not require any transcribed speech, we *do* need some phonetically transcribed speech to interpret the labeling performed by F_1 in order to evaluate its correspondence to linguistically accepted (phonetic) labels; *the transducer F_2 may be considered such an interpreter*. Specifically, we use F_1 to label some held-out speech, i.e speech not used to learn F_1 itself nor to evaluate its efficacy. We use the observed correspondence between the F_1 -derived and the phonetic labels of this held-out speech to learn F_2 , and use it demonstrate 85-90% correspondence between the automatically learned units of F_1 and phonetic transcriptions — in a speaker-dependent setting.

is demonstrated.

The important question of whether the units learned from one speaker generalize to another is addressed in Section 4. We demonstrate that while a labeler (F_1) and interpreter (F_2) learned from one speaker lead to substantially lower accuracies (62-65%) on another speaker *per se*, unsupervised model adaptation techniques such as maximum likelihood linear transforms [8] are able to significantly alleviate the speaker mismatch seen by F_1 ($\sim 75\%$ accuracy). If we further adjust the interpreter (F_2) as well to the new speaker, nearly speaker-dependent performance is achieved (81-84%), clearly suggesting that except for the linguistic interpretation of its labels, which tends to be somewhat speaker-dependent, the acoustic labeling performed by F_1 is substantially speaker-independent.

2. An HMM State-Splitting Algorithm

Takami and Sagayama [5] propose a successive state splitting (SSS) algorithm for learning the topology of a phonemic HMM to capture allophonic variations of the phoneme, and Singer and Ostendorf [7] modify the SSS algorithm to maximize the likelihood of the speech corresponding to all the allophones of the phoneme. We improve the ML-SSS algorithm further, as described below, and use it to directly learn an HMM topology

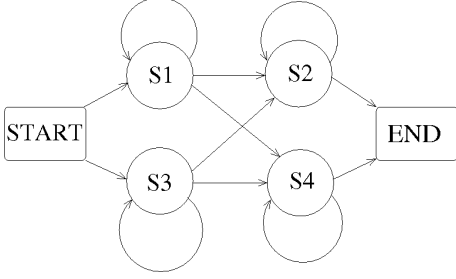


Figure 1: The template for replacing an HMM state s with a quartet of states; START and END are non-emitting states that anchor transitions into and out of state s .

that captures systematic variations in speech. We conjecture that the resulting HMM states will correspond to recurring subword units of speech — phonemes, allophones or subphonetic units — and expect to discover this correspondence by a post hoc analysis of the learned HMM topology in Section 3.

Our modification to ML-SSS algorithm is as follows:

1. Model all speech with a one-state HMM ($N = 1$) with a multivariate Gaussian emission density, and initialize the Gaussian mean μ_1 and covariance Σ_1 from the training data; we use 13-dimensional MFCC coefficients along with the Δ and the $\Delta\Delta$ coefficients.
2. Simultaneously replace each state s of the HMM, $s = 1, \dots, N$, with the 4-state topology shown in Figure 1, yielding a $4N$ -state HMM. If the state s had Gaussian parameters (μ_s, Σ_s) , then means of its 4 replacement states s_1, s_2, s_3 and s_4 are

$$\mu_{s_1} = \mu_s - \delta = \mu_{s_4} \quad \text{and} \quad \mu_{s_2} = \mu_s + \delta = \mu_{s_3}$$

with $\delta = \epsilon\lambda^*v^*$, where λ^* and v^* are the principal eigenvalue and eigenvector of Σ_s and $0 < \epsilon \ll 1$ is typically 0.2. The intuition behind Figure 1 is to simultaneously explore either a contextual or temporal split of s , or *both*, in either order, thereby increasing our search-space by making the search less greedy and potentially yielding greater eventual likelihood.

3. Re-estimate all parameters of this (overgrown) HMM. Gather the Gaussian sufficient statistics for each of the $4N$ states from the last pass of re-estimation: the state occupancy π_{s_i} , sample mean μ_{s_i} and covariance Σ_{s_i} .
4. Each quartet s_1, s_2, s_3, s_4 of states derived from the same original state s can be *merged back* to produce 3 HMM states in 6 different ways, back to 2 HMM states in 7 different ways, and all the back to 1 HMM state, s , in 1 way. For further consideration, retain the best 3-state outcome among the 6 ways, the best 2-state outcome among the 7 ways, and the merge back to a single state, where *best* implies the least loss in likelihood following the merge.
5. Reduce the number of states¹ from $4N$ to $N + \Delta$ by merging back the quartets that cause the *least loss* in total log-likelihood. This entails solving a *constrained knapsack problem*.
6. Set $N = N + \Delta$. If N is less than the desired HMM size, go to Step 2. Otherwise, exit.

¹Typically, $\Delta = \alpha N$ for some $0 < \alpha \ll 1$. When $\Delta = 1$, our algorithm is comparable to ML-SSS; we use full-covariance matrices.

Observe that the 4-state split of Figure 1 permits a slight lookahead in our scheme in the sense that the goodness of a contextual or temporal split of two different states can be compared in the same iteration with two consecutive splits of a single state. Furthermore, the split/merge statistics for a state are gathered in our modified SSS assuming that the other states have already been split, which facilitates consideration of concurrent state splitting. If s_1, \dots, s_m are merged into \tilde{s} , the loss of log-likelihood² in Step 4 is:

$$\frac{d}{2} \sum_{i=1}^m \pi_{s_i} \log |\Sigma_{\tilde{s}}| - \frac{d}{2} \sum_{i=1}^m \pi_{s_i} \log |\Sigma_{s_i}|, \quad (1)$$

where

$$\Sigma_{\tilde{s}} = \frac{\sum_{i=1}^m \pi_{s_i} (\Sigma_{s_i} + \mu_{s_i} \mu_{s_i}^T)}{\sum_{i=1}^m \pi_{s_i}} - \mu_{\tilde{s}} \mu_{\tilde{s}}^T.$$

Finally, in selecting the best Δ states to add to the HMM, we consider all possible ways of splitting the existing N states. e.g. if $N = 6$ and $\Delta = 3$, we consider the best merge-down from $4N = 24$ to $N + \Delta = 9$ states. We want to pick the combination that has the least loss in likelihood. It could be a 4-way split of a single state, a 3-way split of one state and 2-way of another, or a 2-way split of three distinct states. However, each original state s_i is present in the solution, at least with no split, and is not merged with another original state s_j . This restriction leads to an $O(N^5)$ algorithm for what would have otherwise been an incarnation of the 0-1 knapsack problem. The details of the algorithm are omitted for the sake of brevity.

In summary, our modification permits leap-frogging by Δ states at a time compared to the standard ML-SSS algorithm, and benefits from some limited lookahead to avoid greediness.

3. Evaluation of Goodness

The HMM learnt in Section 2 is capable of assigning state labels to speech via the Viterbi algorithm. We view this speech-to-state-label mapping as a finite-state transducer F_1 . Evaluating whether F_1 is labeling the speech with linguistically meaningful units requires mapping the discrete state labels to phoneme labels. We learn a second HMM for this purpose, based on some phonetically transcribed speech, as follows. One may consider this second HMM as a transducer F_2 that *interprets* the labels assigned by F_1 for the purpose of evaluation. To learn F_2 ,

1. We label some held-out speech using F_1 . Generically, for some speech utterance x_1, \dots, x_T ,

$$\hat{s}_1 \dots \hat{s}_T = \arg \max_{s_1 \dots s_T} \prod_{t=1}^T p_{s_t | s_{t-1}} \mathcal{N}(x_t; \mu_{s_t}, \Sigma_{s_t}).$$

We concurrently align the speech with its manual phonetic transcript using standard HMM-based acoustic phonetic models.

2. This permits us to extract the parts of $s_1 \dots s_T$ that align with each *triphone*³ token in the transcript.
3. All F_1 -label sequences that time-align with different realizations of a triphone are pooled together and represented as a finite-state acceptor that accept those and only those label sequences.

²Existence of closed form expression is another advantage of using single-Gaussian densities

³We do this separately for each triphone in light of the fact that the automatically derived units of F_1 are more likely to model realized allophonic units rather than abstract phonemic units of a language.

4. We then perform Bayesian merging of the states of this acceptor using the procedure described by Stolcke et al [9] until the number of states in the acceptor falls below a certain threshold. One such acceptor-of- F_1 -label-sequences is obtained for each triphone.
5. A simple unweighted phone-loop is constructed, and each triphone (i.e. phone in context) is replaced by its corresponding label sequence acceptor described above, resulting in a phone-to-label-sequence transducer F_2 .

The resulting transducer (F_2) may be used to “interpret” label sequences assigned by F_1 to a speech segment via Viterbi decoding. F_2 may also be used as a “language model,” with F_1 in the role of the “acoustic model,” to build a phone-recognizer.

Either is a plausible way of evaluating whether F_1 assigns linguistically meaningful labels to the speech.

4. Cross-speaker Adaptation of Models

In our initial experiments, we derived the HMM labeler F_1 as well as the interpreter F_2 from (disjoint) utterances of a single speaker, say X , and used them to automatically label additional speech from the same speaker X . This results in fairly high phoneme accuracy, as will be presented below in Section 5. However, recognition accuracy degraded significantly when we use F_1 and F_2 learned from the speech of a different speaker Y to label the additional speech of speaker X .

This, however, is not too surprising given that speech recorded from two different speakers varies along many dimensions, some of which may blur the between-phoneme variations within a speaker. We therefore investigate unsupervised adaptation of the transducer F_1 with the intention of compensating for channel and other global variables.

Adapting F_1 to X : The transducer F_1 is adapted to the target (test) speaker X in an unsupervised fashion using some held-out speech. Under the simplifying assumption that background noise and channel variability are the primary differences between the two segments of speech, we divide the HMM state of F_1 into two classes: silence and non-silence. We do a state-level alignment of the HMM states on a small amount of speaker X ’s data. Using this state alignment, we obtain separate linear transforms of the Gaussian means on each of the two classes. E.g., if state s in F_1 had mean μ_s , the new mean of s is $\mathbf{A}_1\mu_s + \mathbf{b}_1$ if it is a “speech state” and $\mathbf{A}_2\mu_s + \mathbf{b}_2$ if it corresponds to silence. \mathbf{A} and \mathbf{b} are estimated in a maximum likelihood fashion.

We use only two regression classes primarily in an attempt to preserve the F_1 -label-to-phoneme mapping, which will be performed by F_2 . If we were to do multi-class regression, it is potentially damaging in the sense that the state-to-phone mapping F_2 may not be preserved, e.g. due to a simple permutation of some similar state labels.

Adapting both F_1 and F_2 to X : Since F_2 may be viewed as merely *interpreting* for evaluation purposes the unsupervised labeling performed by F_1 , it may be argued that F_2 should be learned on speaker X ⁴. If supervised adaptation F_2 to X is permitted, we may be more aggressive in adapting F_1 . To do so, we first construct a regression tree as proposed by Gales [10]. The tree is appropriately cut so that there is enough adaptation speech from speaker X in each leaf. A common MLLR mean-transform is obtained for each leaf using a small amount

⁴This is consistent with notions such as using Levenshtein distance to compute word error rate or BLEU to computer machine translation quality, giving the system maximum possible credit within reason.

of untranscribed speech and its HMM-state level alignment obtained using F_1 . Finer transforms are obtained by repeating the procedure of state-labelling the same untranscribed speech from X and re-estimation. Woodland et al[11] have shown that this iterative approach is beneficial in most cases.

Once the transform-matrices have been estimated, the original HMM F_1 in conjunction with these matrices is used to label some additional (phonetically transcribed) speech from speaker X and F_2 is constructed from it. Recognition of the test speech from X is performed using F_1 , the transforms, and F_2 .

The performance when only F_1 is adapted is indicative of whether F_1 learns sub-word units that generalize meaningfully from speaker Y to speaker X . Only gross variations in the acoustics of X can be normalized by a global MLLR transform, and the labels assigned by the adapted F_1 will be interpreted the same way by F_2 for X as it does for Y .

Adapting both F_1 and F_2 to X is done primarily for comparison with the speaker-dependent performance on X alone.

5. Experiments

5.1. Labeling Consistency on Training Data

We first investigate the extent of the acoustic distinctions that the labeler F_1 needs to preserve in order enable an adequate linguistic interpretation of the F_1 -labeling of the speech as phone sequences. Preserving fine distinctions will require an F_1 with a larger number of states than preserving broad distinctions. In other words, as we vary the number of states in F_1 , we check if the labels assigned to distinct tokens of a phone in the training data are similar to each other and different from labels assigned to other phones. The label sequences⁵ assigned to two instances of the phone u in the context b_N are shown below:

257 [2] 85 [5] 193 [1] 192 [2] 145 [2] and
257 [2] 85 [5] 193 [1] 192 [1] 199 [1] 200 [1] 30 [1].

Although the sequences are not identical, they are very close, and furthermore the two acoustically closest states to state 145 are 199 and 30.

One way to ensure minimal adequacy of F_1 is to make sure that phone accuracy on training data is acceptable! Therefore, we learn F_1 and F_2 from some training speech; we then use the transducer $F_1 \circ F_2$ to recognize the same speech. We present results for different number of states of F_1 in Table 1. It is clear from Table 1 that in a speaker-dependent setting, a point of diminishing returns is reached with about 100 states.

Next, if F_1 were learned from a speaker Y with an adequate number of states to capture phonetic distinctions made by Y , it is conceivable that acoustic distinctions that did not correspond to linguistically meaningful distinctions for Y may matter for a different speaker X , requiring more states in F_1 for speaker-independent acoustic labeling than speaker-dependent.

Thus another way to check if F_1 has an adequate number of states is to label some speech of a speaker, say X , different from the one used to learn F_1 , and to check if a transducer F_2 learned from the F_1 -labeled speech of X has adequate accuracy on its own training data. If F_1 does not make the necessary distinctions in the speech of speaker X , F_2 will not be able to classify its own training data adequately.

Results for this cross-speaker experiment are also presented in Table 1. Again, it should be clear that about 200 states are adequate for cross-speaker labeling. In all subsequent experiments, we use an F_1 with 241 states.

⁵ $s [n]$ denotes an n -length run of the state label s .

# states in F_1	Phone accuracy for speaker-	
	dependent F_1	-independent F_1
17	50.0 %	30.2 %
27	67.2 %	47.0 %
70	94.5 %	84.1 %
106	98.1 %	90.8 %
160	98.1 %	90.3 %
241	99.3 %	97.6 %
376	99.6 %	98.0 %

Table 1: Phone-recognition accuracy of $F_1 \circ F_2$ on the training data of F_2 , shown as a function of the size of F_1 .

5.2. Labeling Accuracy on Test Data

We next investigate the extent to which the acoustic sub-word units learned by F_1 correspond to actual phones, and the extent to which they generalize to new speech from the training speaker, and to speech from a different speaker.

We train an F_1 transducer using untranscribed speech of a speakers, say, A . We then label some additional transcribed speech from A , as well as speech from a different speaker, say, B , using F_1 , and learn two different transducers F_2 , one specialized to each of the two speakers A and B . Finally, we label additional (test) speech from speakers A and B using $F_1 \circ F_2$, and compute the phone accuracy of the labeling.

- (i) Training F_1 on A and F_2 on A , followed by testing on A , represents speaker-dependent performance of F_1 , while
- (ii) Training F_1 on A and F_2 on B followed by testing on B , represents speaker-independent performance of F_1 .

We also experiment with the roles of A and B reversed.

The first two rows of Table 2 represent the phone recognition accuracy of F_1 and demonstrate that when the *interpretation* of the labels (F_2) is speaker-dependent, F_1 generalizes well to the cross-speaker setting: i.e. it makes adequately fine distinctions while still labeling speech consistently enough for F_2 to recover the phone sequence almost as well in both cases.

One would, of course, like even the interpretation of the F_1 -labels to be speaker-independent. There are sufficient differences between recordings of various speakers, however, that under mismatched conditions (e.g. F_1 and F_2 both trained on B and tested on A), phone recognition accuracy drops to 62-64%, as evidenced by the third row in Table 2.

The fourth line of Table 2, in which F_1 is adapted to the test speaker using a global MLLR transform but the interpretation of the F_1 -labels by F_2 is kept unchanged, the accuracy recovers significantly, erasing more than half the errors due to the mismatch. This indicates that global characteristics of the speech, such as channel distortions are a significant factor, and if they are factored out, labels provided by F_1 on held-out speech of an unseen speaker correspond to phones approximately 75% of the time.

Finally, as a sanity check in the last row of Table 2, where we adapt a mismatched F_1 to the test speaker *as well as* learn the interpretation of the F_1 -labels, i.e. F_2 , from additional speech from the test speaker, the phone recognition accuracy almost catches up with the speaker-dependent levels. This is expected, and merely validates the procedures used.

6. Conclusions

We have demonstrated that acoustic subword units may be learned reasonably effectively by our modified ML-SSS algorithm proposed in Section 2. Furthermore, we have quantified

F_1 training + adaptation data	F_2 training data	Test	
		$X = A$	$X = B$
X	X	89.0 %	85.5 %
Y	X	84.2 %	81.0 %
Y	Y	62.0 %	61.0 %
$Y + X_{\text{MLLR}(1)}$	Y	74.2 %	72.2 %
$Y + X_{\text{MLLR}(n)}$	X	87.4 %	83.1 %

Table 2: Phone-recognition accuracy of automatically learned subword units. X designates the same speaker in training and testing; Y designates a speaker mismatch.

the extent to which the learned units generalize to new utterances of the training speaker (very well) and to new utterances from other speakers (quite well). Some work remains to identify improvements to the algorithm that may further close the gap between speaker-dependent and speaker-independent performance, perhaps with techniques such as speaker adaptive training with multiple training speakers.

7. References

- [1] T. Svendsen and F. Soong, "On the automatic segmentation of speech signals," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '87.*, vol. 12, 1987, pp. 77,80.
- [2] K. Paliwal and A. Kulkarni, "Segmentation and labeling using vector quantization and its application in isolated word recognition," pp. 102-110, 1987.
- [3] F. Jelinek, *Statistical Methods for Speech Recognition*. The MIT Press, 1998.
- [4] Y. Shiraki and M. Honda, "Lpc speech coding based on variable-length segment quantization," pp. 1437,1444, 1988.
- [5] J. Takami and S. Sagayama, "A successive state splitting algorithm for efficient allophone modeling," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, 1992.
- [6] Fukada, Bacchiani, Paliwal, and Sagisaka, "Speech recognition based on acoustically derived segment units," vol. 2, 1996, pp. 1077-1080.
- [7] H. Singer and M. Ostendorf, "Maximum likelihood successive state splitting," vol. 2. ICASSP, May 1996, pp. 601,604.
- [8] M. Gales, "Maximum likelihood linear transformations for hmm-based speech recognition." Tech. Report, CUED/FINFENG/TR291, Cambridge Univ., 1997., 1997.
- [9] A. Stolcke and S. Omohundro, "Hidden Markov model induction by Bayesian model merging," *Advances in Neural Information Processing Systems*, vol. 5, pp. 11-18, 1993.
- [10] M. Gales, "The generation and use of regression class trees for mlr adaptation," 1996.
- [11] P. Woodland, D. Pye, and M. Gales, "Iterative unsupervised adaptation using maximum likelihood linear regression," in *Proc. ICSLP '96*, vol. 2, Philadelphia, PA, 1996, pp. 1133-1136.