Joint Lexicon, Acoustic Unit Inventory and Model Design

M. Bacchiani and M. Ostendorf

Boston University

Electrical and Computer Engineering Department

8 St. Mary's Street

Boston, MA 02215

30 August 1998

List of unusual symbols: All mathematical symbols used are standard or are defined in the text.

Number of pages: 31, including cover page, this page and lists of tables and figures

Number of tables: 4

Number of figures: 2

Keywords: lexicon design, acoustic model clustering, pronunciation modeling

Joint Lexicon, Acoustic Unit Inventory and Model Design

M. Bacchiani M. Ostendorf

Boston University

30 August 1998

Abstract

Although most parameters in a speech recognition system are estimated from data by use of an objective function, the unit inventory and lexicon are generally hand crafted and therefore unlikely to be optimal. This paper proposes a joint solution to the related problems of learning a unit inventory and corresponding lexicon from data. On a speaker-independent read speech task with a 1k vocabulary, the proposed algorithm outperforms a phone-based system at low complexity and performs equivalently or better at high complexity.

Zusammenfassung

Obwohl die meisten Parameter eines Spracherkennungssystems aus Daten geschätzt werden, ist die Wahl der akustischen Grundeinheiten und des Lexikons normalerweise nicht automatisch und deshalb wahrscheinlich nicht optimal. Dieser Artikel stellt einen kombinierten Ansatz für die Lösung dieser verwandten Probleme dar – das Lernen von akustischen Grundeinheiten und des zugehörigen Lexikons aus Daten. Experimente mit sprecher-unabhängigen gelesenen Sprachdaten mit einem Vokabular von 1000 Wörtern zeigen, daß der vorgestellte Ansatz besser ist als ein System niedriger Komplexität, das auf Phonemen basiert ist, und besser oder vergleichbar für Systeme höherer Komplexität.

Résumé

Bien que la plupart des paramètres dans un système de reconnaissance de la parole soient estimés à partie des données en utilisant une fonction objective, l'inventaire des unités acoustiques et le lexique sont généralement créés a la main, et donc susceptibles de ne pas être optimeux. Cette étude propose une solution conjointe aux problèmes interdépendants que sont l'apprentissage à partir des données d'un inventaire des unités acoustiques et du lexique correspondant. Nous avons testé l'algorithme proposé sur des échantillons lus, en reconnaissance indépendantes du locuteur avec un vocabulaire de 1k: il surpassé les systèmes phonétiques en faible complexité, et a fait au moins aussi bien en forte complexité.

1 Introduction

Large vocabulary speech recognition systems typically represent lexical entries in terms of sub-word units, for which acoustic models can be reliably estimated. Part of the system design is therefore to decide on a suitable unit inventory and define the lexicon, i.e. define the mappings from lexical entries in the vocabulary to linear strings or networks of units. This problem is simplified in most systems by using phone-based units and a hand-crafted lexicon, frequently with a single linear phoneme string for the majority of the lexical entries. Although the parameters of the unit models are generally estimated from data using an objective function such as maximum likelihood (ML), no such function is used in the unit inventory and lexicon design. Given the lack of a clear objective in this part of the system design, the resulting unit inventory and lexicon are unlikely to be optimal in terms of the objective function used throughout the design of the rest of the system.

Even though the unit inventory and lexicon definition is suboptimal, the use of mixture distributions in the unit models has proven successful in read speech tasks. When this system design algorithm is applied to spontaneous speech tasks, however, severe performance degradation compared to read speech tasks is observed. Where mixture distributions in

the unit models for read speech are able to capture the acoustic variability within the sub-optimally defined units, the increased acoustic variability in spontaneous speech makes this approach problematic because of the increased confusability between units.

An alternative to manual derivation of a unit inventory and lexicon is to learn them from data. A unit derived in this way is generally referred to as an acoustic sub-word unit (ASWU). Over the last decade, a number of researchers have looked into this problem and found algorithms that automatically define model inventories and estimate unit model parameters (Lee *et al.,* 1989; Svendsen *et al.,* 1989; Bahl *et al.,* 1993; Bacchiani *et al.,* 1996). The related problem of defining a lexicon in terms of these ASWUs has also received attention, e.g. (Paliwal, 1990; Svendsen, Soong & Purnhagen, 1995). To derive a lexicon of linear string pronunciations, all these approaches start by finding a number of candidate pronunciations for each lexical entry in the vocabulary based on the pronunciations seen across training tokens for the target lexical entry. These candidate pronunciations are then evaluated with all training tokens, and an objective function is used to determine the optimal candidate.

One problem with this approach is that it decouples the unit inventory and lexicon design problems, which are clearly related. As a result, acoustic models are no longer optimal after the pronunciations are restricted. This problem can be addressed by iterative re-estimation of the acoustic model and the pronunciations, as in (Holter & Svendsen, 1997). A bigger problem, not addressed in previous work, is that for tasks with a lot of pronunciation variability in the initial labeling of tokens, it is expensive to determine the optimal pronunciation among the large number of candidates and difficult to rule out cases when the vast majority occur only once. Furthermore, the optimal pronunciation may not even be among the observed candidates. Cases where a large number of candidate pronunciations per lexical entry can be expected include speaker-independent recognition, where "pronunciation" variability would be a consequence of speaker differences, and/or large vocabulary recognition where a large inventory of acoustic units (analogous to polyphonic HMM states)

is needed.

Here we describe a joint unit inventory and lexicon design algorithm that addresses the problem of initial pronunciation variability by introducing lexical constraints into the unit inventory design. By designing the units and lexicon jointly, the derived units and their associated acoustic models are matched to the lexicon. Section 2 describes the algorithm in detail. In section 3, experimental results on the DARPA Resource Management corpus are described. Finally, the open research issues are discussed in section 4.

2 Algorithm

The two basic steps of any unit inventory design algorithm are an acoustic segmentation step followed by a clustering step, e.g. (Lee *et al.,* 1989; Svendsen *et al.,* 1989; Bacchiani *et al.,* 1996; Holter & Svendsen, 1997). In most systems, lexicon design would be a subsequent step, with the goal being to find a single linear "pronunciation" for each word. Similarly, our focus is on deriving a single linear pronunciation for each word[1], but *within* the inventory design process. More specifically, the key elements that differ in our approach compared to previous work are the use of pronunciation-related constraints in unit design, the consistent use of a maximum likelihood objective function, and progressive unit inventory and model refinement.

Using the linear pronunciation assumption, two important new constraints are introduced that allow joint inventory and lexicon design. First, the segmentation step is constrained to use the same number of segments for every token of a particular lexical entry. Then, the clustering step is constrained by pre-grouping all the segments in the different training tokens of a particular lexical entry according to their position in the fixed-length sequence. The lexicon is implicitly defined after completion of the clustering step, since the data from different training tokens representing the same position within a lexical entry are assigned to a single cluster. In addition, since the maximum likelihood objective function

is used, the acoustic model parameters are also defined as a result of clustering. Section 2.1 describes segmentation and clustering with constraints in more detail.

Progressive unit refinement is important for at least three reasons. First, once data is clustered, the segmentation that the initial units were based on may no longer be appropriate. Section 2.2 describes how the unit model inventory and corresponding segmentation can be refined further using retraining to obtain a better match between the segmentation and the models. Second, even phone-based systems benefit from progressive techniques for increasing the acoustic model complexity, both in terms of contextual and temporal resolution (Woodland & Young, 1993; Takami & Sagayama, 1992; Ostendorf & Singer, 1997). Some solutions for ASWUs are described in Sections 2.3 and 2.4, respectively.

2.1 Initial unit inventory and lexicon design

As mentioned above, the initial inventory and lexicon design is a two-step process, involving segmentation and then clustering. The first step provides a segmentation in which all training tokens of a lexical entry contain the same number of segments. In the second step, the segments are clustered subject to a pronunciation consistency constraint to define the unit inventory and lexicon. The segmentation and clustering algorithms can be implemented for polynomial mean trajectory segment models in general (Bacchiani *et al.,* 1996; Kannan & Ostendorf, 1998), but for simplicity the experiments and equations given here correspond to the special case of a hidden Markov model, i.e. a constant mean trajectory.

The first step in designing an ASWU system is **acoustic segmentation**, that is, finding segmentation times that divide each word token into piecewise stationary regions that can be reasonably well modeled with a single HMM state. In our case, the number of regions per token is fixed for a particular lexical item – a pronunciation length constraint. The pronunciation length could be specified in terms of some baseline phone pronunciation length, but some of the potential power of automatic learning is lost in this case. Instead, we first run an unconstrained acoustic segmentation, then choose the median number of

segment associated with a lexical item for the length constraint, and finally rerun acoustic segmentation subject to the length constraint, as described below.

Unconstrained acoustic segmentation functions as an initialization step. Taking an approach similar to that in (Paliwal, 1990), the maximum likelihood segmentation of the training data is found by use of dynamic programming. Let x_t be a d-dimensional observation vector, such as a vector of cepstral coefficients representing a window of speech at time t. The unconstrained acoustic segmentation algorithm involves recursive updating for every time t and every allowable number of segments n:

$$\delta(t,n) = \max_{t-l_{max} \leq \tau \leq t-l_{min}} \left[\delta(\tau - 1, n-1) + \log \mathrm{p}(x_\tau, \ldots, x_t | \mu_{\tau,t}, \Sigma) \right], \tag{1}$$

where l_{min} and l_{max} denote minimum and maximum segment lengths. In addition to updating $\delta(t,n)$, the index τ that maximizes equation (1) is stored, allowing the most likely segmentation to be found in the end by tracing back. The (generalized) likelihood of the segments during the dynamic programming is computed using a multivariate Gaussian model with a single diagonal covariance Σ used for all the segments. (This covariance matrix can be estimated either on a per utterance basis, a per speaker basis, or from the entire training corpus.) The mean parameter of the Gaussian model is computed from the hypothesized segments; the constant mean model corresponds to the assumption that speech is piecewise stationary.

During segmentation, the likelihood increases monotonically with the number of allowable segments, since there are more free parameters with which to fit the data. A thresholding mechanism is used to control the average number of segments, defining the temporal resolution (average state duration) of the resulting system. To control the average number of segments, we investigated using an average-likelihood-per-frame threshold L_{avg}, as well as a weighted minimum description length (MDL) criterion. In the weighted MDL case, the number of segments $\hat{S}$ is determined by

$$\hat{S} = \operatorname*{argmin}_S -\delta(T,S) + \frac{\alpha}{2} m(S) \log(T) \tag{2}$$

where T denotes the number of feature vectors in the utterance and $m(S)$ denotes the number of free parameters used in the segmentation S. The penalty term is weighted by α to allow some external control over the temporal resolution of the system, so as to have comparable complexity with the two methods. Both the threshold L_{avg} and MDL weight α are chosen empirically to obtain an average segment length close to what is associated with a 3-state per phone system for the target recognition task.

Next, the acoustic segmentation is aligned with automatically-derived word segmentation times by assigning each acoustic segment to the lexical-entry token with which it overlaps most. (The word segmentation times are given by forced alignment to the word transcriptions using a context-dependent HMM system.) For each lexical entry, the median of the number of acoustic segments over all the training tokens is used to define the pronunciation length. The training data is then segmented again using dynamic programming (eqn. 1) under the constraints that:

- the number of acoustic segments for a lexical item is equal to the median pronunciation length, and
- each lexical entry boundary coincides with an acoustic segment boundary.[2]

In the resulting segmentation, all training tokens of a lexical entry have the same number of segments.

The second step involves **clustering** the results of the segmentation step to define the unit inventory. The clustering algorithm used here differs from that used in (Svendsen *et al.,* 1989; Paliwal, 1990; Holter & Svendsen, 1997) in that maximum likelihood is used as an objective rather than minimum Euclidean distance. Specifically, the repartitioning step involves computing the likelihood of segments given the model parameters of a cluster, i.e. the "distance" is a negative log likelihood. The cluster re-estimation procedure consists of finding the ML parameter estimates of a Gaussian distribution from the data contained in the cluster. Cluster centroids therefore directly represent unit models and clustering

addresses both the inventory and model design problems, whereas in (Svendsen *et al.,* 1989; Paliwal, 1990; Holter & Svendsen, 1997) unit model parameters had to be estimated in a separate step from the data partition defined by clustering.

Before clustering, the data is grouped to ensure pronunciation consistency, our second pronunciation constraint. Using the fact that clustering is based on a maximum likelihood objective, the grouping is implemented by computing a sufficient statistic for each collection of segments originating from different training tokens in the same position within a lexical entry. The sufficient statistics for the constant mean model are the sample mean μ_p and covariance Σ_p and the total number of vector observations contained within the group N_p. These statistics are stored for each unique position within each unique lexical entry: if there are V entries in the vocabulary and the average median pronunciation length is R, the data is grouped into VR groups. These statistics will be referred to as *atomic group sufficient statistics.* As the sufficient statistic representations of these atomic groups cannot be split in clustering, this grouping ensures pronunciation consistency.

Let a group of K segment observations $\mathcal{X}_p = \{X^1, \cdots, X^K\}$, of lengths $\{L_1, \cdots, L_K\}$, have sufficient statistics (μ_p, Σ_p, N_p) where $N_p = \sum_{i=1}^{K} L_i$. The distance of the group with respect to the cluster with parameters μ_c and Σ_c is computed as the negative log likelihood

$$-\mathcal{L}(\mathcal{X}_p|\mu_c, \Sigma_c) = \frac{N_p}{2}\left[D\log(2\pi) + \log(|\Sigma_c|) + \mathrm{tr}\left(\Sigma_c^{-1}\Sigma_p\right) + (\mu_p - \mu_c)'\Sigma_c^{-1}(\mu_p - \mu_c)\right], \quad (3)$$

where $'$ denotes transposition. Once observations are assigned to a cluster, the ML parameter estimate given P groups of observations are

$$\mu_c = \frac{1}{\sum_{q=1}^{P} N_q} \sum_{p=1}^{P} N_p \mu_p \quad (4)$$

$$\Sigma_c = \frac{1}{\sum_{q=1}^{P} N_q} \left[\sum_{p=1}^{P} N_p \left(\Sigma_p + \mu_p\mu_p' - \mu_c\mu_c'\right)\right]. \quad (5)$$

Starting from a single cluster, the cluster inventory is increased by binary divisive clustering. Iteratively, the cluster with the lowest average likelihood per frame is selected to be

split. Two new clusters are defined by first obtaining an initial split by perturbing the cluster mean, and then running a number of binary K-means clustering iterations using only the data that was contained in the original cluster before splitting. Clusters with fewer observations than a minimum occupancy threshold (100) are not considered for splitting. After the cluster inventory is increased up to a heuristically determined inventory size, a number of K-means iterations (10) using all the data and the full cluster inventory are run. If any cluster during this stage has fewer observations than the minimum occupancy threshold, the cluster is removed from the inventory and the data previously held within that cluster is repartitioned over the remaining clusters. The clustering algorithm derives the final unit model inventory by alternating between divisive and K-means iterations, increasing the number of clusters in stages. Final inventory sizes are chosen to be similar to that used in the comparable HMM systems. Note that inventory sizes are difficult to tightly control, since they are a reduced version of the specified target due to the elimination of clusters falling below the minimum occupancy threshold.

The final data partition over the cluster inventory defines the lexicon by virtue of the data grouping. When neighboring segments within a lexical entry are assigned to the same cluster, the segments are collapsed into a single entry in the lexicon. As the units assume a stationary distribution, repetitions of the same unit label in the lexicon are equivalent to a single instance of the label. (Note that segments that were found to be distinct in unrestricted acoustic segmentation can be labeled as equivalent after the quantization introduced by clustering.) During this unit collapsing step, some temporal resolution is lost relative to the temporal resolution of the acoustic segmentation. Options to control the temporal resolution of the system are described in section 2.4.

2.2 Re-training

The acoustic segmentation was optimal given an unconstrained model inventory (size S for S segments), since model parameters in acoustic segmentation are derived separately

for each instance of a segment in the dynamic programming search. After clustering, the acoustic space is quantized into C models with $C << S$, making the acoustic segmentation suboptimal. To achieve a matched condition between the unit model inventory and the segmentation, a retraining algorithm can be used, either Viterbi or Baum-Welch. The Viterbi training algorithm, used here, iteratively re-segments the data to find the optimal segmentation given the current unit model inventory and then re-estimates unit model parameters using the new segmentation. Given a lexicon consisting of linear pronunciation strings derived by the algorithm described in section 2.1, a lexical-entry-level transcription can be expanded into an S-length unit-level transcription $\{u_1, \cdots, u_S\}$. For the general case that allows segment models, the segmentation step involves recursive updating for every time t and every unit index $i \in \{1, \cdots, S\}$ of

$$\Psi(t,i) = \max_{1 \leq \tau \leq t-1} \Psi(\tau - 1, i - 1) + \log \mathrm{p}(x_\tau, \cdots, x_t | \mu_{c(i)}, \Sigma_{c(i)}) \tag{6}$$

where $\mu_{c(i)}$ and $\Sigma_{c(i)}$ denote the mean and covariance of the i-th segment which has unit label $c(i)$. In addition to updating $\Psi(t,i)$, the index τ that maximizes equation (6) is stored, allowing the most likely segmentation to be found in the end by tracing back. For the HMM, this equation can be simplified to a forced alignment using the Viterbi algorithm. The unit model parameters are re-estimated from this new segmentation using standard ML estimation.

2.3 Increasing system complexity: larger unit inventory

For most recognition tasks, HMM systems tend to give improved performance as complexity is increased, in the sense of having a larger unit inventory (e.g. monophones vs. triphones vs. quinphones). These larger inventories are typically obtained using automatic clustering of models with different phonetic contexts, e.g. (Young & Woodland, 1993). One way to obtain a high complexity ASWU system is to obtain a large unit inventory by a single divisive clustering run, starting from the segment boundaries derived by acoustic segmentation.

Alternatively, one could run Viterbi training after an intermediate size unit inventory is designed, re-estimate the atomic group sufficient statistics, and continue divisive clustering to increase the inventory using these new statistics. In preliminary experiments, we found that better results were obtained using this second approach.

Once one has the intermediate size inventory and new sufficient statistics, the system complexity can be increased in several ways. Three alternatives are investigated here, as described next. In all three cases, the inventory is increased by iterative application of ML clustering (divisive followed by K-means) and Viterbi training.

At each ML clustering step, the unit inventory is increased by use of divisive clustering until either a heuristically determined inventory size is reached or the average likelihood per frame exceeds a threshold (empirically set to achieve a desired inventory size) or all cluster occupancies fall below the minimum occupancy threshold. Then ten K-means iterations are run, removing clusters that have fewer observations than the minimum occupancy, as described in Section 2.1. After Viterbi re-segmentation, models are re-estimated and, if a larger inventory is desired, new atomic group sufficient statistics are computed based on the new segment boundaries for use in a subsequent clustering step. The key differences between the three high-complexity system variations are the definition of the atomic units and the imposition of constraints on clustering of those units.

- **High complexity ASWU system.** In the first option, the atomic group sufficient statistics are defined in terms of lexical position, as for the intermediate system but the statistics are different because of the resegmentation.

- **Unconstrained CD-ASWU system.** Another approach to increase the complexity of an ASWU-based system is by estimating parameters of context-dependent ASWU (CD-ASWU) models. In this case, atomic group sufficient statistics are computed for each unit in each unique left and right context, i.e. defined in terms of local context rather than word position.

- **Center-constrained CD-ASWU system.** In the third approach, atomic group sufficient statistics are again computed for each unit in each unique left and right context, but parameter sharing is limited to models with the same center unit.

The CD-ASWU systems are motivated by the success of explicitly modeling context in a phone-based system. This center-constrained version of CD-ASWU is roughly equivalent to the approach used in phone-based systems where only parameter sharing among the same state of context-dependent models with the same center phone are allowed.

2.4 Increasing system complexity: temporal resolution refinement

The iterative ML clustering and Viterbi training approach is beneficial as it allows retaining near-optimal segment boundaries in the unit inventory design algorithm. A problem it introduces however is that throughout the unit design, the temporal resolution of the system decreases. When neighboring units within a lexical entry are clustered in the same cluster, they are merged to form a single unit causing loss of temporal resolution. Note that segments merged given a small unit inventory might not have been merged given a large unit inventory, so it may be useful to allow segment splits for the high complexity system.

One approach to avoid this problem is to set the threshold that controls the temporal resolution in the first unconstrained acoustic segmentation so that initially segments are very short (high temporal resolution), compensating for the loss in temporal resolution incurred during the unit inventory design algorithm. A problem with this approach is that the resulting system has lower accuracy, probably because of poor decisions on parameter sharing during the clustering stage. Another approach to circumvent the problem of the loss of temporal resolution is to increase the temporal resolution by splitting the segments derived after a Viterbi training stage. Each individual segment in each word token is split in two using the acoustic segmentation algorithm described in section 2.1. New atomic group sufficient statistics are then computed for the new segmentation, and a new lexicon can be

defined by running one or more K-means clustering iterations, starting from the existing unit model inventory. Successive identical units will be merged as before, so the effective increase in pronunciation length is much less than a factor of two.

3 Experiments

Experiments were conducted on the DARPA Resource Management corpus (Price *et al.*, 1988), which is a read speech corpus with a 991 word vocabulary. Although the proposed algorithm is developed with the ultimate goal of attacking the increased acoustic variability problem in spontaneous speech corpora, initial experiments were conducted on the smaller Resource Management corpus to investigate the viability of the algorithm and explore different options at a lower experimental cost.

The 109-speaker training set of approximately 228 minutes of speech was used as training material for unit inventory and lexicon design. The training set contains 46814 training tokens for 991 unique words. The most frequently observed word has 4112 training tokens, the least frequent has 1 training token. The average number of training tokens per word is 47, the median 11. The February 1989 test set containing 300 utterances was used as a development test set. The four available test sets (Feb 89, Oct 89, Feb 91 and Sept 92) were used in the main system comparisons to have more reliable results, and the average of the four results will be referred to as the *full test set* performance. The word-pair grammar provided with the Resource Management corpus was used in the search.

Feature vectors were computed every 10 ms and included 12 Mel-warped cepstral coefficients and normalized energy and their first and second order differences (39 dimensions in total). The speech signal was windowed using a Hamming window of 25 ms, and a first-order pre-emphasis filter (0.97) was applied. Diagonal covariances were used throughout the experiments.

The word recognition performance is derived from a dynamic programming word-level

string alignment of the recognizer output to the reference transcription. Denoting the number of correct labels as H, the number of insertions as I and the total number of labels in the reference transcription as N, the percent correct figure is defined as $H/N \times 100\%$, and the percent accuracy figure is defined as $(H - I)/N \times 100\%$.

3.1 Phone systems

For performance comparisons, phone-based HMM systems were trained using the HTK toolkit (HTK, 1997). The 48-phone lexicon provided with the Resource Management corpus were used. Except for the silence model which used a single state, the 48 phone HMMs had a 3 state left-to-right topology without allowing skips. Starting from context-independent (CI) model parameter estimates derived from the TIMIT corpus, 4 Baum-Welch training iterations were performed to derive a 145 state CI system. These models were then cloned for each unique phone context (expanded to a triphone system), and 2 Baum-Welch training iterations were run to train the 2254 model/6760 state system. Two parameter sharing techniques were used on the 2254 triphone models. One involved agglomerative clustering, resulting in 1514 distributions, and the other used tree-based clustering, which gave 1100 distributions. Both clustering systems outperformed the unclustered triphone system, roughly 90% accuracy for the clustered systems compared to 87.5% accuracy for the unclustered triphones on the February 1989 test set.

The recognition performance on the February 1989 and full test sets is given in table 1. As expected, the clustered triphone systems significantly outperform the context-independent system. The agglomerative clustering approach gave slightly better results than tree-based clustering, but this may simply reflect the larger number of distributions in the agglomerative system. This difference in size is a consequence of differences in the agglomerative vs. divisive clustering strategies, as well as HTK control parameters for determining tree size.

(Location of Table 1)

3.2 ASWU systems

To derive an ASWU unit model inventory and lexicon, the algorithm described in section 2 was applied. The thresholding at the acoustic segmentation stage was set so that the resulting segmentation contained approximately 3 acoustic segments per phone, which gives a temporal resolution comparable to the phone-based systems. Preliminary experiments using coarser and finer segmentations showed performance degradation. During each ML clustering stage, the minimum cluster occupancy was set to 100 frames; clusters with fewer observations were removed. The process of removing infrequent clusters often resulted in a significant decrease in inventory size, so that it was sometimes difficult to obtain inventories of the same size when comparing different ASWU design methods.

The first experiment was with a **low complexity system**, for comparison to the HMM context-independent phone models. Starting from an unconstrained acoustic segmentation based on average-likelihood-per-frame thresholding with a grand covariance, a 141 unit system was designed using alternating divisive and K-means clustering, with Viterbi retraining at sizes 124 and 141. The performance on the February 1989 and full tests sets was 80.1% and 78.2%, respectively, which compares favorably to the accuracy of the CI phone system.

All experiments with high complexity systems used **progressive refinement** to obtain a series of unit inventories of different sizes. An example of progressive refinement is illustrated in Figure 1, which shows how performance improves with increasing the inventory and Viterbi resegmentation. In this particular experiment series, the initial unconstrained acoustic segmentation was based on average-likelihood-per-frame thresholding with a covariance estimated separately for each utterance. An initial inventory of size 124 was designed using 5 stages of alternating divisive and K-means clustering, followed by 3 iterations of Viterbi training. Then, the inventory was increased first to 342 units and then to 646 units, using 5 and 7 stages of alternating divisive and K-means clustering, respectively and

2 iterations of Viterbi training at each target inventory size. Next, the temporal resolution was increased by splitting all segments in the final Viterbi segmentation in two and re-clustering, as described in section 2.4. This reduced the inventory size to 635 units after removing clusters with low counts (< 100). A final divisive plus K-means clustering stage increased the inventory to 1385 units.

(Location of Figure 1)

To investigate the **options for thresholding and variance estimation** in the acoustic segmentation step, three low complexity systems (containing between 124 and 126 units) were trained and then evaluated on the February 1989 test set. Two systems used a covariance estimated on a per utterance basis. Of these two systems, one used the average-likelihood-per-frame thresholding and one used the weighted MDL measure. The third system also used an average-likelihood-per-frame threshold but used a covariance estimated from the complete training set (grand covariance). For all systems, the unit model inventory and corresponding lexicon was derived in 5 stages of alternating divisive and K-means clustering, followed by 3 iterations of Viterbi training. The performance ranged from 75.6% accuracy for the MDL-based system to 76.8% accuracy for both the average-likelihood-per-frame thresholded systems. Thus, we discontinued unconstrained acoustic segmentation with the MDL criterion. The unit inventory for the two average-likelihood-per-frame threshold initial segmentations – per-utterance covariance and grand covariance – were then increased to 646 and 631 units, respectively. The performance of the grand covariance based system was 84.2% accuracy compared to an accuracy of 85.4% for the per-utterance variance system. Since the per-utterance variance estimation seemed to have a slight advantage at this level, the remaining experiments use that technique.

In another experiment series, we looked at the effect of changing the point of **temporal splitting** in the progressive refinement steps. The 646 unit inventory was expanded, without temporal splitting, to a size of 1147 units by an additional ML clustering and Viterbi training

step. The temporal resolution was then increased by splitting the segments in two through acoustic segmentation, and 5 iterations of K-means clustering were run. The resulting inventory, which included 1098 units after low frequency clusters were removed, was then refined using 2 passes of Viterbi training. These two systems were compared to the system described earlier, illustrated in figure 1, that introduced temporal refinement at an earlier stage and resulted in a 1385 unit inventory. The results for the three alternatives on the February 1989 test set are given in table 2, showing that some sort of temporal refinement is important, but that the particular stage within progressive refinement is not important. The performance of the 1098 and 1385 unit systems on the full test set is 88.2% and 89.3% accuracy, respectively. Both systems have an average of 2-3 automatically derived units per phone in the lexical representation, so the temporal resolution is just a bit coarser than an HMM state.

(Location of Table 2)

A number of lexical entries in the vocabulary have the same morpheme bases, such as *alert* and *alerts*. Even though the pronunciation of the morpheme base can vary as a function of the affix due to effects such as coarticulation, some consistency in pronunciation can be expected. However, since the pronunciation constraints are applied independently for each word, there is no guarantee of pronunciation consistency across words. As one would hope, experiments showed that in many cases the resulting pronunciations are in fact quite similar, particularly for the low complexity systems. Some examples of lexical entries with equal morpheme bases and their pronunciation in terms of units from the low complexity system are given in table 3.

(Location of Table 3)

3.3 CD-ASWU systems

Three CD-ASWU experiments were conducted to test the effect of unconstrained vs. constrained clustering and size of the base inventory. The base inventory (either 124 or 635 units) was expanded by considering "tri-units" to be units conditioned on the unit label of their left and right neighbor. Atomic group sufficient statistics were computed for these context-dependent units for use in subsequent clustering. Depending on the size of the base inventory, the number of atomic groups (observed tri-units) was 6.3k vs. 13k. The 6.3k atomic groups were used for designing both an unconstrained and a center-constrained CD-ASWU system; the 13k atomic groups were used only in an unconstrained CD-ASWU system. As in previous experiments, inventory size was increased using progressive refinement, and clusters with less than 100 observations were eliminated in all ML clustering steps. As an example, figure 2 illustrates the performance associated with progressive steps in designing the unconstrained CD-ASWU system from 6.3k atomic groups. (In this case, which is atypical, Viterbi training actually hurt performance for the largest inventory.)

(Location of Figure 2)

The resulting recognition performance of the three different cases with the associated inventory size are given in table 4, in comparison to the the ASWU system that defines the atomic groups in terms of lexical position rather than tri-unit context. (The 1385-unit ASWU system is used, since that system starts from the same 635-unit system used to design the tri-unit context for the second unconstrained CD-ASWU system.) Although the final inventory size is larger for the unconstrained case, extrapolating from the performance of the unconstrained case, there appears to be no advantage to center-constrained clustering and possibly some performance degradation. We also find that increasing the number of atomic groups leads to improved performance for the unconstrained CD-ASWU system. Since having more atomic groups gives more degrees of freedom in clustering, the two results can be interpreted more generally as suggesting that fewer constraints on clustering are

beneficial. Finally, we observe that lexical position dependence gives better performance, 90.1% vs. 89.3% accuracy, but this may be explained by the larger degrees of freedom in clustering by having 30k vs. 13k atomic groups. To test this hypothesis, we would need to define quin-unit atomic groups for CD-ASWU clustering. Quin-unit cluster would also come closer to triphone modeling: since units here are analogous to HMM states and not whole phones, the tri-unit context window is effectively smaller than for triphones.

(Location of Table 4)

4 Discussion

In summary, the proposed algorithm approaches the problems of automatic unit and lexicon design as a joint problem ensuring a matched condition between unit models and the lexicon. As in previous work, a unit inventory is designed by an acoustic segmentation step followed by a clustering step. A joint solution is obtained by constraining the unit design algorithm to guarantee a limited complexity of the pronunciation model. Two constraints are imposed in the design, with the objective of finding a single linear pronunciation per word. First, a pronunciation length constraint enforces uniformity in the number of segments across training tokens of a lexical entry. Second, a pronunciation consistency constraint ensures that segments in the same word position of different word tokens are assigned the same unit label. A final unit inventory and lexicon are designed by progressive refinements, alternating expansion of the unit inventory by clustering with adjustment of the segmentation by Viterbi training. The temporal resolution of the system can optionally be refined in the re-segmentation steps.

Comparing the performance of the proposed automatic unit inventory and lexicon design algorithm at low complexity, the 141 ASWU unit system outperforms the 145 state phone-based system. The 124-unit ASWU system performs comparably to the 128-unit system described in (Holter & Svendsen, 1997) and to the 145-state CI phone-based HMM system.

At high complexity, the performance of the ASWU systems with 1098 and 1385 states is comparable to the triphone systems with 1100 and 1514 states, respectively. Since the ASWU system is in fact an HMM, further improvements in performance can be obtained by using mixture distributions, as in (Woodland & Young, 1993).

Several variations of the ASWU system were explored, yielding three main conclusions. First, the use of an average-likelihood-per-frame thresholding approach in the acoustic segmentation stage performed comparably or better than using weighted MDL thresholding, and the use of a per-utterance estimated covariance gave slightly better performance than a grand covariance estimate. Second, using fewer constraints in the unit inventory and lexicon design algorithm when increasing the complexity of the system results in better performance. Finally, for both the unconstrained ASWU as well as the CD-ASWU systems, increased temporal resolution is important for better system performance, though the particular stage in inventory design for increasing temporal resolution did not appear to be an important factor.

A problem of the proposed algorithm is that it requires several training tokens for each lexical entry. For a large vocabulary system, this is an unrealistic requirement. One option for addressing this problem is to design a hybrid system that uses phonetic units for rare (i.e. low frequency) lexical entries and automatic units for those entries having "sufficient" training tokens (Bacchiani & Ostendorf, 1998). In order to understand how much data is "sufficient" for ASWU pronunciation modeling, the best case system was analyzed to determine the relation between training token counts and two different indicators of pronunciation "goodness". First, the relation between the percentage correspondence between pairs of lexical entries with the same morpheme base (such as the examples in Table 3) and training token count was modeled. Second, the training token count as a function of the relative error in the February 1989 test set was modeled. A linear model gave a good fit for both relationships (correlation of 0.97 and 0.85 for the two regression models respectively), and the data did not show a clear breakpoint for use as a training token count thresh-

old. Thus, the trade-off point between phonetic and automatically-derived units should be assessed experimentally.

Another possible limitation of the algorithm described here is the assumption of a single linear pronunciation per lexical item. Certainly the use of mixture distributions, which can be incorporated after the initial unit/lexicon design step, can compensate for some pronunciation variability, as in phone-based HMM systems. However, it may be useful to explicitly represent pronunciation variants, including effects of cross-word phonetic context and phonological variation. Within-word pronunciation variation can be successfully modeled with ASWUs, as demonstrated by Holter and Svendsen (1998). For our approach, a straightforward extension of the temporal unit splitting algorithm to contextual unit splitting would allow for multiple pronunciations within the context of unit design with pronunciation constraints, assuming a redefinition of atomic units.

The comparable performance of the low complexity system with the results of (Holter & Svendsen, 1997) demonstrates that the constrained unit design approach is competitive with previous ASWU work. The performance at high complexity, comparable to a triphone system, shows the success of the algorithm for the design of large unit inventories, which is an important advance on previous work. Given that phone-based systems represent the state-of-the-art in read speech tasks, especially at high system complexity, the comparable performance of the ASWU-based system shows the viability of the proposed algorithm. However, since the ASWU approach currently relies on the availability of word segmentation times (which are typically provided by a phone-based system), it is important to demonstrate a significant gain over phone-based systems. The hope is that, in spontaneous speech recognition tasks, the ASWU-based system will perform better than a phone-based system using standard baseforms, since speakers use canonical pronunciations much less frequently in casual speech.

Acknowledgments

This work was supported by ATR Interpreting Telecommunications Laboratory.

Footnotes

1. Since single-pronunciation dictionaries are relatively successful for many speech recognition tasks, particularly for first-pass decoding stages of a multi-pass search, we defer the problem of representing pronunciation variation within words for the future (see Section 4).

2. In a small number of cases, the median segment length is longer than the number of speech frames in a token, making segmentation impossible, in which case the word boundary times are relaxed.

References

[1] M. Bacchiani, M. Ostendorf, Y. Sagisaka, and K. Paliwal (1996). "Design of a speech recognition system based on non-uniform segmental units," in: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Vol. 1, pp. 443-446.

[2] M. Bacchiani and M. Ostendorf (1998). "Using automatically-derived acoustic subword units in large vocabulary speech recognition," *Proceedings of the International Conference on Spoken Language Processing,* to appear.

[3] L.R. Bahl, P.F. Brown, P.V. de Souza, R.L. Mercer, and M.A. Picheny (1993), "A method for the construction of acoustic Markov models for words," *IEEE Transactions on Speech and Audio Processing,* Vol. 1, No. 4, pp. 443-452.

[4] T. Holter and T. Svendsen (1997). "Combined optimisation of baseforms and model parameters in speech recognition based on acoustic subword units," in: *Proceedings of the IEEE Workshop on Automatic Speech Recognition,* pp. 199-206.

[5] T. Holter and T. Svendsen (1998). "Maximum likelihood modeling of pronunciation variation," in: *Proceedings of the ESCA Workshop on Modeling Pronunciation Variation for Automatic Speech Recognition,* pp. 63-66.

[6] HTK, Version 2.1, Cambridge University, 1997.

[7] A. Kannan and M. Ostendorf (1998), "A comparison of constrained trajectory models for large vocabulary speech recognition," *IEEE Transactions on Speech and Audio Processing,* Vol. 6, No. 3, pp. 303-306.

[8] C.H. Lee, B.-H. Juang, F.K. Soong, and L. Rabiner (1989). "Word recognition using whole word and subword models," in: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Vol. 1, pp. 683-686.

[9] M. Ostendorf and H. Singer (1997), "HMM topology design using maximum likelihood successive state splitting," *Computer Speech and Language,* Vol. 11, No. 1, pp. 17-42.

[10] K.K. Paliwal (1990). "Lexicon building methods for an acoustic sub-word based speech recognizer," in: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Vol. 2, pp. 729-732.

[11] P. Price, W.M. Fisher, J. Bernstein, D.S. Pallett (1988). "The DARPA 1000-Word Resource Management database for continuous speech recognition," in: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Vol. 1, pp. 651-654.

[12] T. Svendsen, K.K. Paliwal, E. Harborg, and P.O. Husøy (1989). "An improved subword based speech recognizer," in: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Vol. 1, pp. 108-111.

[13] T. Svendsen, F.K. Soong, and H. Purnhagen (1995). "Optimizing baseforms for HMM-based speech recognition," in: *Proceedings European Conference on Speech Communication and Technology,* Vol. 1, pp. 783-786.

[14] J. Takami and S. Sagayama, (1992). "A successive state splitting algorithm for efficient allophone modeling," in: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing,* Vol. 1, pp. 573-576.

[15] P.C. Woodland and S.J. Young (1993). "The HTK tied-state continuous speech recogniser," in: *Proceedings European Conference on Speech Communication and Technology,* Vol. 3, pp. 2207-2210.

[16] S.J. Young and P.C. Woodland (1993). "The use of state tying in continuous speech recognition," in: *Proceedings European Conference on Speech Communication and Technology,* Vol. 3, pp. 2203-2206.

List of Figures

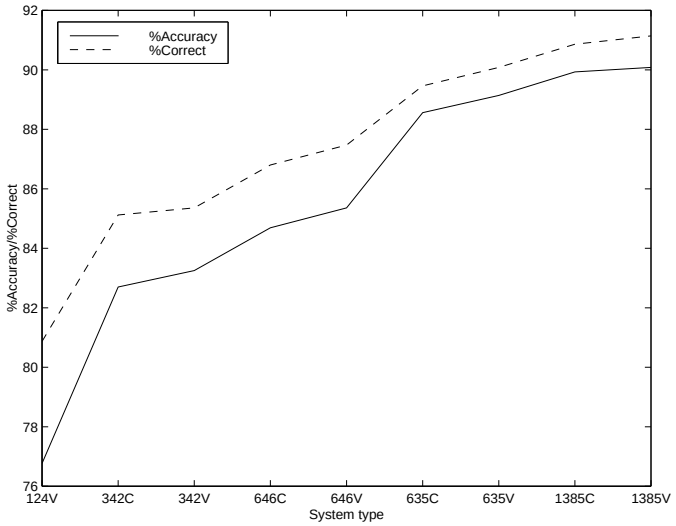

Figure 1: Recognition performance on the February 1989 test set using unit inventories and lexicons derived at different stages of progressive refinement. System type indicates the number of units and is appended with either a C for the system after clustering or a V for the system after Viterbi training.

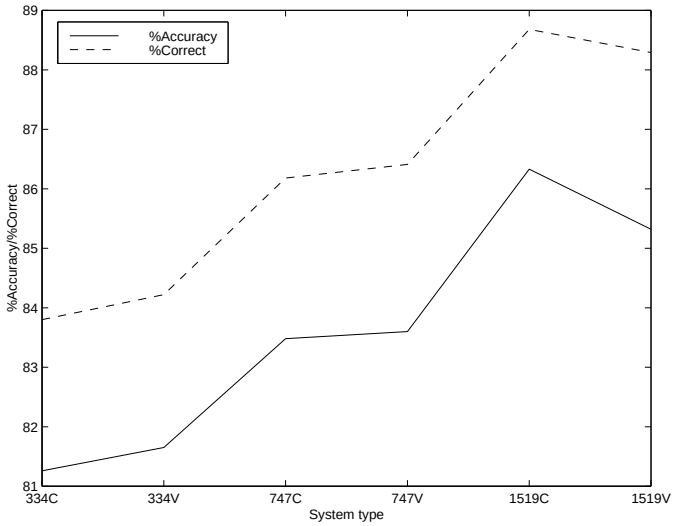

Figure 2: Recognition performance on the February 1989 test set using unit inventories and lexicons derived at different stages of the unconstrained CD-ASWU training algorithm. System type indicates the number of units and is appended with either a C for the system after clustering or a V for the system after Viterbi training.

List of Tables

System	Number of states	Accuracy	
		Feb 89 Test	Full Test
Context Independent	145	75.6	75.6
Agglomerative clustered triphones	1514	90.2	89.1
Tree-based clustered triphones	1100	89.8	88.6

Table 1: HMM system results (% accuracy) on the February 1989 and full test sets for phone-based different system configurations.

System	Refinement Strategy	% Accuracy
No temporal refinement	646 $\rightarrow$ 1147	86.9
Early temporal refinement	646 $\rightarrow$ split (635) $\rightarrow$ 1385	90.1
Late temporal refinement	646 $\rightarrow$ 1147 $\rightarrow$ split (1098)	90.4

Table 2: Recognition performance on the February 1989 test set for different temporal refinement approaches.

ALERT	U9	U104		U47	U115	U21	U112	U82	U91	U100	U125
ALERTS	U9	U104	U25	U47	U115		U112	U82	U91	U100	...
LETTER	U79	U47	U115	U66	U104	U87	U103	U71	U70		
LETTERS	U79	U47	U115	U66	U104	U87	U103	U84	U70	...	
NINETEEN	U123		U33	U115	U66		U133	U57	U129	U109	...
NINETEENTH	U123	U36	U33	U115	U66	U75	U133	U57	U129	U109	...
NINETY	U123		U33	U115	U66		U133	U12	U123	U20	...

Table 3: Pronunciation examples of lexical entries with the same morpheme base.

Refinement Strategy	Atomic Groups	Inventory Size	% Accuracy
Center-constrained CD-ASWU	6.3k	1262	84.8
Unconstrained CD-ASWU	6.3k	1519	86.3
Unconstrained CD-ASWU	13k	1382	89.3
Lexical position ASWU	30k	1385	90.1

Table 4: Recognition performance on the February 1989 test set for different approaches to increasing the unit inventory size.