STATE-BASED GAUSSIAN SELECTION IN LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION USING HMMS

M.J.F.Gales[1], K.M.Knill[2] & S.J.Young

January 1997 (revised January 1998)

EDICS Number: SA 1.6.10

Cambridge University Engineering Department,

Trumpington Street, Cambridge CB2 1PZ, England

Email: mjfg@watson.ibm.com, kmk@nuance.com, sjy@eng.cam.ac.uk

Abstract

This paper investigates the use of Gaussian Selection (GS) to increase the speed of a large vocabulary speech recognition system. Typically 30-70% of the computational time of a continuous density HMM-based speech recogniser is spent calculating probabilities. The aim of GS is to reduce this load by selecting the subset of Gaussian component likelihoods that should be computed given a particular input vector. This paper examines new techniques for obtaining "good" Gaussian subsets or "shortlists". All the new schemes make use of state information, specifically which state each of the Gaussian components belongs to. In this way a maximum number of Gaussian components per state may be specified, hence reducing the size of the shortlist. The first technique introduced is a simple extension of the standard GS method, which uses this state information. Then, more complex schemes based on maximising the likelihood of the training data are proposed. These new approaches are compared with the standard GS scheme on a large vocabulary speech recognition task. On this task, the use of state information reduced the percentage of Gaussians computed to 10-15%, compared with 20-30% for the standard GS scheme, with little degradation in performance.

[1]M.J.F.Gales is now at the IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA.

[2]K.M. Knill is now at Nuance Communications, 1380 Willow Rd, Menlo Park, CA 94025, USA.

List of Tables

1 Introduction

High accuracy large vocabulary continuous speech recognition (LVCSR) HMM-based systems have been developed in recent years. These systems tend to operate at several times real-time which is not practical for most applications. Techniques are therefore required to reduce the decoding time to faster than real-time, while maintaining, or staying close to, the same level of accuracy.

To obtain a high level of accuracy, LVCSR HMM-based systems typically use continuous density HMMs (CDHMMs). In such systems, calculation of the state likelihoods makes up a significant proportion (between 30 to 70%) of the computational load. This is a result of the need to use multiple mixture Gaussian output distributions in a state. Anywhere from 8 to 64 Gaussian components are typically used and each Gaussian component must be separately evaluated in order to determine the overall likelihood. A wide variety of techniques may be used to reduce the amount of computation required. Most of these techniques involve alteration of the acoustic feature vector and/or the acoustic models from the full CDHMM system. For example, linear discriminant analysis [Hunt and Lefebre, 1989] can be used to reduce the number of elements in the feature vector, or the Gaussian components may be more tightly "tied" as in a semi-continuous HMM system [Huang et al., 1990]. The problem with these techniques is that, in the majority of cases, these modifications lead to a degradation in performance. An alternative approach is to use Gaussian Selection (GS) [Bocchieri, 1993]. GS methods reduce the likelihood computation of a system by only computing the likelihood of a selected subset, or "shortlist" [Murveit et al., 1994], of Gaussian components at each frame. The underlying system remains unchanged. This paper presents techniques to optimise the selection of the shortlists within the GS framework to minimise the likelihood computation without degrading recognition accuracy.

The motivation behind GS is as follows. If an input vector is an outlier with respect to a Gaussian component distribution, i.e. it lies on the tail of the distribution, then the likelihood of that Gaussian component producing the input vector is very small. This results in the likelihood of the input frame per Gaussian component within a state having a large dynamic range, with one or two Gaussian components tending to dominate the state likelihood for a particular input vector. Hence, the state likelihood could be computed solely from these Gaussian components without a noticeable loss in accuracy. GS methods attempt to efficiently select these Gaussian components, or a subset containing them, for each input vector. A number of different methods have been proposed which can be classified in terms of their approach to partitioning the acoustic space, including; vector quantisation (VQ) [Bocchieri, 1993, Murveit et al., 1994], binary search trees [Fritsch and Rogina, 1996], dynamic searches [Beyerlein, 1994, Beyerlein and Ullrich, 1995] or a combination of these [Ortmanns et al., 1997].

This paper addresses the question of how to select the "best" shortlist given a division of acoustic space. The best shortlist can be defined as the one that minimises the likelihood computation for no, or minimal, loss in recognition accuracy. This work was based on the original GS scheme proposed by Bocchieri [Bocchieri, 1993]. However, it is directly applicable to any VQ-based GS scheme and similar methods may be applied to any static GS approach. The range of GS techniques is described in section 2. Section 3 reviews the standard GS scheme of Bocchieri, how the Gaussian codewords are generated and how performance is assessed. The following section describes how state information may be incorporated into the selection process. Section 5 details a refinement of this based on a maximum likelihood approach to GS. These schemes are evaluated on a standard large vocabulary speech recognition task, the unlimited vocabulary Hub 1 task from the 1994 ARPA evaluation [Pallett et al., 1995].

2 Gaussian Selection Techniques

In the original implementation of GS by Bocchieri [Bocchieri, 1993] the acoustic space is divided up during training into a set of vector quantised regions. Each Gaussian component is then assigned to one or more VQ codewords. The subset of Gaussians linked to a codeword is commonly called the shortlist [Murveit et al., 1994]. During recognition, the input speech vector is vector quantised, that is the vector is mapped to a single VQ codeword. The likelihood of each Gaussian component in this codeword's shortlist is computed exactly. For the remaining Gaussian components, the likelihood is approximated. The VQ codebooks can be generated by clustering either the Gaussian components [Bocchieri, 1993] or training data [Murveit et al., 1994].

Alternatively acoustic space can be partitioned by building binary search trees and assigning Gaussian components relative to the tree nodes. One such technique is the Bucket Box Intersection (BBI) [Fritsch and Rogina, 1996]. In each dimension of acoustic space there is a region where the log probability of a Gaussian component exceeds some threshold. The regions associated with a particular Gaussian component are enclosed within a K-dimensional box. This allows rapid partitioning of the acoustic space. The acoustic space is partitioned by building a K-d (K dimensions, depth d) binary tree. This produces 2^d disjoint rectangular regions (buckets). The tree is built so that the splits between regions minimise the number of bucket-box intersections. In recognition, an input vector can be assigned to a unique bucket by descending the tree. Only Gaussians whose box's intersect with the current bucket are computed. Multiple trees are built for different groups of Gaussian components. This type of scheme is best suited to systems with a large number of Gaussian components per state (or mixture).

If the HMM system has a single shared covariance then fast dynamic searches can be implemented during recognition to determine which Gaussian components are close to the input vector [3].

[3] On-line distance metric based searches are not suited to systems without a shared covariance as the computation required to perform the search is close to the full likelihood computation.

Fast nearest neighbour searches take advantage of the fact that Gaussian components can be eliminated from likelihood computation based on a partial or relative (to other Gaussian components) distance. The exact distance between the Gaussian component and the input vector therefore does not have to be computed so approximations can be used in the search. For example the Recall-Jump-Eliminate and partial-distance algorithms [Beyerlein, 1994] and the Hamming distance approximation [Beyerlein and Ullrich, 1995]. Alternatively, a tree can be built dynamically such as in the projection search algorithm (PSA) [Ortmanns et al., 1997]. In the PSA, acoustic space is gradually partitioned to leave a small hypercube centered around the input vector. The number of Gaussian component likelihoods to be fully evaluated is reduced at each partition by eliminating the Gaussian components that lie outside the new hypercube. Unlike the K-d tree, a dimension may be revisited. These techniques can be combined with each other or with VQ shortlists [Ortmanns et al., 1997].

3 VQ-Based Gaussian Selection

The first task in VQ-based Gaussian selection is to generate a set of codewords, or clusters. Each codeword is then assigned a shortlist of Gaussians. When an input vector is mapped to a codeword during recognition, the likelihood of each Gaussian in the associated shortlist is computed. The decrease in computational load is dependent on the size of this shortlist.

3.1 Gaussian Cluster Generation

For the work presented here a single codebook was used. The codewords were generated by clustering Gaussians during training as follows. A weighted Euclidean distance between the means was used to calculate the distance between the ith and jth Gaussians.

$$\delta(\mu_i, \mu_j) = \frac{1}{K}\sum_{k=1}^{K}\left\{w(k)(\mu_i(k) - \mu_j(k))\right\}^2 \tag{1}$$

where K is the feature vector dimension, $\mu_i(k)$ is the kth component of vector μ_i, and $w(k)$ is equal to the inverse square root of the kth diagonal element of the average of the covariances of the Gaussian set $\mathcal{N}(\mu_m, \mathbf{\Sigma}_m), m = 1, \ldots, M$, where $\mathbf{\Sigma}_m$ is the diagonal covariance of the mth Gaussian component. Using this metric, a clustering procedure based on the Linde-Buzo-Gray [Linde et al., 1980] algorithm was applied to minimise the average (per Gaussian component) distortion, δ_{avg},

$$\delta_{avg} = \frac{1}{M}\sum_{m=1}^{M}\left\{\min_{\phi=1}^{\Phi}\delta(\mu_m, \mathbf{c}_\phi)\right\} \tag{2}$$

where M is the number of Gaussian components in the model set, Φ is the number of clusters (pre-defined), and $\mathbf{c}_\phi$ is the centre (codeword) of the ϕth cluster χ_ϕ,

$$\mathbf{c}_\phi = \frac{1}{size(\chi_\phi)} \sum_{m \in \chi_\phi} \mu_m, \phi = 1, \ldots, \Phi \tag{3}$$

In all schemes, it is necessary to assign each observation vector (i.e. speech input vector) to one of the clusters during recognition in order to select the Gaussian shortlist to evaluate. The appropriate cluster is selected by determining the cluster centroid, $\mathbf{c}_\phi$, which minimises the weighted Euclidean distance to the observation vector, $\mathbf{o}(t)$, at time t,

$$\chi_t = \arg\min_{\chi_\phi} \delta(\mathbf{o}(t), \mathbf{c}_\phi) \tag{4}$$

In some of the Gaussian selection schemes proposed here it is also necessary to assign the training data to one of the clusters. Although the same cluster selection scheme may be used, there is also the option of using a softmax assignment. Hence for generality when describing the various Gaussian selection schemes, the cluster selection will be described in terms of a "probability", $p(\chi_\phi|\mathbf{o}(t))$. However, in all experiments described in this work a hard assignment as determined by equation 4 was actually used.

3.2 Standard Gaussian Selection

If a Gaussian shortlist for cluster χ_ϕ, ν_ϕ, was generated solely from the Gaussians used in the cluster generation, the clusters would be disjoint. When used in recognition, errors would then be likely since the likelihood of some Gaussians close to the input feature vector but not in the selected cluster would be excluded from the full computation. To avoid this, clusters share Gaussians as follows. Given a threshold $\Theta \geq 0$, an input feature vector, $\mathbf{o}$, is said to fall on the tail of the mth Gaussian if

$$\frac{1}{K} \sum_{k=1}^{K} \frac{(o(k) - \mu_m(k))^2}{\sigma_m^2(k)} \gg 1 \tag{5}$$

where $\sigma_m^2(k)$ is the ith diagonal element of $\mathbf{\Sigma}_m$. Thus, if the cluster centroid is taken to be a typical input feature vector, the Gaussian shortlist, ν_ϕ, of codeword χ_ϕ, can be defined as consisting of all Gaussians such that [Bocchieri, 1993]

$$G(m) \in \nu_\phi \;\text{ iff }\; \frac{1}{K} \sum_{k=1}^{K} \frac{(c_\phi(k) - \mu_m(k))^2}{\sigma_{avg}^2(k)} \leq \Theta \tag{6}$$

where $\sigma_{avg}^2(i)$ is the ith diagonal element of the matrix of the average covariance of all Gaussians in the HMM set and $G(m)$ indicates Gaussian component m of the standard recognition system.

An alternative method for determining which Gaussians are to be in the shortlist is to use

$$G(m) \in \nu_\phi \;\text{ iff }\; \frac{1}{K} \sum_{k=1}^{K} \frac{(c_\phi(k) - \mu_m(k))^2}{\sqrt{(\sigma_{avg}^2(k)\sigma_m^2(k))}} \leq \Theta \tag{7}$$

The advantage of this measure is that Gaussian components with large variances are encouraged to be assigned to a cluster and this has been shown to perform better than the scheme described by equation 6 [Knill et al., 1996] [4]. This scheme, which is referred to as *Standard Gaussian Selection* (SGS), is used here to provide a performance baseline.

3.3 Gaussian Selection Performance

The performance of the various Gaussian selection schemes is assessed in terms of both the recognition performance and the reduction in the number of Gaussian components calculated. This reduction, the computation fraction, C, is defined as

$$C = \frac{G_{new} + VQ_{comp}}{G_{full}} \tag{8}$$

where G_{new}, G_{full} are the average number of Gaussians calculated per frame in the GS and full system respectively, and VQ_{comp} the number of computations required to calculate the VQ index. The latter is dependent on the size of the codebook and search technique used. A number of techniques exist to efficiently select the codeword (such as [Poggi, 1993]) so the contribution of VQ_{comp} to the computation fraction is minimal. For these reasons, in this paper performance is assessed only in terms of the number of Gaussians calculated compared to the standard recognition system, that is $VQ_{comp} = 0$ in equation 8.

4 State-based Gaussian Selection

The choice of Θ in equation 7 controls the average size of the Gaussian shortlist. Efficiency improves with reductions in Θ because fewer Gaussian likelihoods have to be computed during recognition. However, there is a trade-off with the recognition accuracy. For states with at least one Gaussian component assigned to a selected cluster, errors can occur if some Gaussian components that make a significant contribution to a state likelihood are not contained in that cluster. Further errors can occur due to 'state flooring' which occurs when no Gaussian components from a state are assigned to the selected cluster. In this case the state likelihood is simply given a discrete approximate value. Since it is possible for an input vector to be an outlier with respect to all the Gaussian component distributions of a state on the optimal search path, the state flooring chosen can be crucial to maintaining accuracy.

A good cluster assignment is therefore one which assigns all, or most of, the Gaussian components that contribute significantly to state likelihoods for that region of acoustic space, while assigning as few non-contributing Gaussian components as possible. It is not possible to satisfy this "good" clustering requirement with the previous schemes, as no account was made of which

[4]Equation 7 shows the distance measure used for this work and the work described in [Knill et al., 1996] (there termed the class-weighted distance measure). The equation given in [Knill et al., 1996] is incorrect.

state the Gaussian component belonged to. The concept of retaining the Gaussian component to state information is therefore introduced (based on preliminary work in [Knill et al., 1996]). This scheme will be referred to as *State-Based Gaussian Selection* (SBGS). SBGS allows a maximum number of Gaussian components associated with each state to be defined using the following selection routine

$$G(jm) \in \nu_\phi \ \text{ iff } \ G(jm) \in \arg\min_m(n)\left\{D(G(jm), \mathbf{c}_\phi)\right\} \tag{9}$$

where $G(jm)$ is the Gaussian component m of state j, $D(G(jm), \mathbf{c}_\phi)$ represents the "distance" from $G(jm)$ to the cluster centroid $\mathbf{c}_\phi$ and for this work is the same measure as used in equation 7, and $\arg\min_m(n)\{\}$ represents selecting the minimum n Gaussian components according to the selected distance measure. There is also an added constraint that $D(G(jm), \mathbf{c}_\phi) \leq \Theta$.

The state-based selection process can be further refined if the following assumption is made. That is, the Gaussian components closest to the cluster centroid are more likely to produce the data that will be mapped to that codeword. Under this assumption, those states with Gaussian components near to the centre of the cluster should be modelled more accurately by assigning more Gaussian components from these states. This is essentially equivalent to assigning more Gaussian components from states which are often seen in the training data for that codeword and fewer to those seen less. This requirement is most easily achieved using a "multi-ring" approach. In this approach, two extra thresholds are applied, Θ_1, Θ_2, where $\Theta_1 < \Theta_2 \leq \Theta$. If the distance of the closest Gaussian component of state j to the cluster centroid $\mathbf{c}_\phi$ is less than Θ_1 then the maximum number of Gaussian components in the shortlist associated with that state is set to N_1. The second threshold Θ_2 is then used to give a maximum number of Gaussian components for states whose closest Gaussian component falls between Θ_1 and Θ_2. This is simply described by the following algorithm for selecting the maximum number of Gaussian components n in equation 9

if $(\min_m \{D(G(jm), \mathbf{c}_\phi)\} \leq \Theta_1)$ then

$n = N_1$

else if $(\min_m \{D(G(jm), \mathbf{c}_\phi)\} \leq \Theta_2)$ then

$n = N_2$

else

$n = 0$

where $N_1 \geq N_2 \geq 0$, with the constraint throughout that for all selected Gaussian components $D(G(jm), \mathbf{c}_\phi) \leq \Theta$. Thus Θ_2 controls which states are to be "floored" and Θ_1 controls which states are more accurately modelled. This is a simple two-ring approach, though of course more complex functions may be used.

5 Maximum Likelihood Gaussian Selection

5.1 Selection Process

Gaussian selection aims to select the set of Gaussians that will best model the data that will be assigned to a particular cluster during recognition. The GS techniques described above base this selection purely on distance from the cluster centroid. This is clearly sub-optimal since it requires two crude assumptions to be made. First, the centroid is assumed to be representative of the acoustic realisations of each Gaussian associated with that cluster. However, since Gaussians from many different phonetic contexts will usually be assigned to the same cluster, the associated data frames may differ substantially. Thus, the cluster centroid may be a poor representation of the data from a particular context. Second, the choice of Gaussians takes no account of their ability to model the data. For example the actual distribution may be bimodal, with one peak near the center of the cluster and the second further out. Simply choosing the closest Gaussian components to the cluster center will not well model the data in this case.

To overcome these problems an alternative scheme, *Maximum Likelihood Gaussian Selection* (MLGS) is proposed. The aim of this scheme is to select the set of Gaussians that minimise the difference in likelihood of the training data between the standard and Gaussian selected systems[5]. Thus, the Gaussian shortlist for state j and cluster χ_ϕ, $\hat{\nu}_\phi$, is selected according to

$$\hat{\nu}_\phi = \arg\min_{\nu_\phi} \left\{ \sum_{t=1}^{T} \gamma_j(t) p(\chi_\phi|\mathbf{o}(t)) \log\left(\sum_{m=1}^{M_j} w_{jm}\mathcal{N}(\mathbf{o}(t);\mu_{jm},\mathbf{\Sigma}_{jm}) \right) - \sum_{t=1}^{T} \gamma_j(t) p(\chi_\phi|\mathbf{o}(t)) \log\left(\sum_{G(jm)\in\nu_\phi} w_{jm}\mathcal{N}(\mathbf{o}(t);\mu_{jm},\mathbf{\Sigma}_{jm}) \right) \right\} \quad (10)$$

where M_j is the number of Gaussian components in state j and the state occupation probability $\gamma_j(t)$ is given by

$$\gamma_j(t) = p(q_j(t)|\mathbf{O}_T) \quad (11)$$

where $q_j(t)$ denotes the occupation of state j at time t. This probability is computed from the system with no GS. Hence the basic criterion for selecting Gaussian components for the shortlist is to choose the set that minimises the loss in likelihood between the standard no GS system and the new GS system with the assumption that the frame-state alignments do not significantly alter.

Given that Gaussian components may only be left out, the Gaussian shortlist that maximises the probability of the training data, will be the one that minimises equation 10[6]. To find the "true" best set of Gaussian components requires checking through all possible combinations of the required number of Gaussians. The ordering of the groups of Gaussian components is then based

[5]These are not really likelihoods, but auxiliary function values. It is therefore assumed for this work that the frame-state alignment will be the same for both the full and Gaussian selected scheme.

[6]This assumes that a well trained initial model set is used.

on the likelihood scores. This may be done, but for many systems quickly becomes impractical[7]. Appendix A describes an algorithm which closely approximates this desired scheme.

For practical applications, however, a simpler implementation strategy is desirable. If it is assumed that for a state-cluster pairing, the associated data is well modelled by the n Gaussian components from that state with the highest likelihood, then the selection process becomes

$$G(jm) \in \nu_\phi \text{ iff } G(jm) \in \arg\max_m(n) \left\{ \sum_{t=1}^{T} \gamma_j(t) p(\chi_\phi|\mathbf{o}(t)) \log\left(w_{jm}\mathcal{N}(\mathbf{o}(t); \mu_{jm}, \mathbf{\Sigma}_{jm})\right) \right\} \quad (12)$$

Although this approximate scheme is only guaranteed to yield the maximum likelihood solution when a single Gaussian component is to be selected, it is computationally very efficient. It also requires relatively little memory, since it is only necessary to store the probability of the state-cluster pairing for each Gaussian component-cluster pairing. It is this implementation of maximum likelihood GS that is used in this work and is referred to as MLGS.

A simple alternative scheme, not based on likelihoods, which attempts to improve the modelling of the data from the state-cluster pairing is to use the occupancy counts. Here

$$G(jm) \in \nu_\phi \text{ iff } G(jm) \in \arg\max_m(n) \left\{ \sum_{t=1}^{T} \gamma_{jm}(t) p(\chi_\phi|\mathbf{o}(t)) \right\} \quad (13)$$

where the component occupation probability $\gamma_{jm}(t)$ is given by

$$\gamma_{jm}(t) = p(q_{jm}(t)|\mathbf{o}_T) \quad (14)$$

and $q_{jm}(t)$ indicates "being in" Gaussian component m of state j at time t. Again this probability is calculated using the standard, no GS, system. This selects the set of Gaussian components with the greatest occupancy for that cluster-state pairing. The example of bimodal distributions is therefore well handled as the Gaussian components "nearest" the two maxima should have higher occupancy counts. Unfortunately in this scheme the data from Gaussian components with low occupancy counts is completely ignored when selecting the Gaussian components for the shortlist. This will be referred to as *Occupancy Gaussian Selection* (OGS).

5.2 Handling Limited Training Data

It has so far been implicitly assumed that there is an infinite amount of training data. Thus all state-cluster pairs that will ever be feasible will have non-zero occupancies. Of course in reality this is not the case. For most large vocabulary systems the number of Gaussian components and states is selected to make best use of the available training data. It is therefore very unlikely that all feasible pairings will be observed. However what can be stated is that the most likely pairings are liable to occur in the training data.

[7] Suppose that 4 Gaussian components are to be selected from each 12 Gaussian component state. In this case, there are 495 ($\frac{12!}{4!8!}$) combinations to consider for each state-cluster pairing. This also assumes that it is known a-priori the number of Gaussian components required for the particular pairing.

In most systems the vast majority of the occupancies for these state-cluster pairings will be zero. Additionally many of the remaining pairings will have very low counts making them very poor estimates of the true[8] data associated with the state-cluster pairing. Hence basing a selection process on such state-cluster pairings may result in a "poor" shortlist. However the problem of robust estimates is a standard one in speech recognition. Many of the standard techniques used for robustly building large vocabulary systems may be used here, for example appendix B describes how decision trees may be used for this task.

In the work presented here, however, a simple backing-off scheme is used whereby each state-cluster pairing is only used without any modification if there is "sufficient data". At this most specific level, the context-dependent level, the determination of "sufficient data" is simply based on a minimum occupancy measure, l_d. For those states exceeding this threshold the selection process is the same as equation 12. Unfortunately many state-cluster pairings will have low occupancy counts and for these pairings it is necessary to back-off to the context-independent level. If all contexts of each phone have the same HMM topology, then each state may be separately considered at the context-independent level. For MLGS with backoff, the shortlist selection function defined by equation 12 is thus modified as follows

$$G(jm) \in \nu_\phi \;\text{ iff }\; G(jm) \in \arg\max_m(n) \left\{ \sum_{k \in \mathbf{s}(j)} \sum_{t=1}^{T} \gamma_k(t) p(\chi_\phi|\mathbf{o}(t)) \log\left(w_{jm}\mathcal{N}(\mathbf{o}(t);\mu_{jm},\mathbf{\Sigma}_{jm})\right) \right\} \tag{15}$$

where $\mathbf{s}(j)$ indicates the list of all states associated with the same monophone as state j and in the same state position. Again a minimum occupancy threshold, l_i, may be used to determine whether there is sufficient data at this stage. If there is still insufficient data it is necessary to back-off to the cluster level. Here all data associated with a particular cluster is used in the selection process. At this stage there is little useful temporal information in the state position, so all the states are combined together. The selection process now becomes

$$G(jm) \in \nu_\phi \;\text{ iff }\; G(jm) \in \arg\max_m(n) \left\{ \sum_{t=1}^{T} p(\chi_\phi|\mathbf{o}(t)) \log\left(w_{jm}\mathcal{N}(\mathbf{o}(t);\mu_{jm},\mathbf{\Sigma}_{jm})\right) \right\} \tag{16}$$

The shortlist generated using equation 16 is the same as that which would be produced if SBGS were used with a likelihood distance measure. It is not possible to sensibly back-off any further.

This simple backing-off scheme has been described in terms of the MLGS scheme. It is not appropriate for the OGS scheme as it would be necessary to select which of the Gaussian components in the state to assign the occupancy of the additional backed-off data. This may be done, using either of the assignment schemes described in appendix A, but is not considered here. In this work a partial OGS scheme was implemented. OGS was used to select Gaussian components in states that satisfied the context-dependent threshold, l_d. For the remaining states, the MLGS

[8]In the sense of the data associated with an infinitely large training database.

scheme was used.

5.3 Number of Gaussians Per Codeword

The SBGS scheme uses a varying number of Gaussians for each state depending on how "close" the nearest Gaussian component for a state was to the cluster centroid. A similar scheme is used in MLGS. When considering the total likelihood of the training data, rather than one state at a time, it is logical to assign more Gaussian components to commonly occurring states than less commonly occurring ones. This may be calculated more explicitly in the MLGS scheme (than SBGS) as the state-cluster occupancy counts may be calculated. Hence, the number of Gaussian components may now be directly related to the state-cluster occupancies. However there is a problem with this approach. For state-component pairings that seldom occur the occupancy counts will be very "noisy" and as such a poor measure of which states to "floor" completely. For the experiments carried out here state flooring was determined simply by how "close" a Gaussian was to the cluster centroid.

For the MLGS and the partial OGS schemes with backing-off described above the maximum number of Gaussians from a state per codeword was determined in the following way. For each state-cluster pairing, state j and cluster χ_ϕ,

if $(\sum_{t=1}^{T} \gamma_j(t) p(\chi_\phi|\mathbf{o}(t)) > l_d)$ then

$n = N_1$

else if $(\sum_{k\in\mathbf{s}(j)} \sum_{t=1}^{T} \gamma_k(t) p(\chi_\phi|\mathbf{o}(t)) > l_i)$ then

$n = N_2$

else if $(\min_m \{D(G(jm), \mathbf{c}_\phi)\} \leq \Theta)$

$n = N_3$

else

$n = 0$

This is a very simple scheme for selecting the number of Gaussians required. If more complex schemes are used, such as those described in appendix A, it is possible to explicitly calculate the contribution of each Gaussian component to the likelihood score of the training data.

In common with all other GS schemes, there is the question of how to deal with the floored states and, to a lesser extent, the floored Gaussian components. For all the experiments carried out in this paper a simple fixed value was used for all the floored Gaussian components. To obtain the contribution of each floored Gaussian component to the state likelihood, this value is scaled by the Gaussian component weight. Hence the likelihood of a floored state is the same fixed value. The floor value used in this work was not optimised for the specific task chosen.

6 Experiments and Results

6.1 System Descriptions

The HTK standard large vocabulary speech recognition system was used to compare the performance of the various GS schemes. This was configured as a gender-independent cross-word-triphone mixture-Gaussian tied-state HMM system identical to the "HMM-1" model set used in the HTK 1994 ARPA evaluation system [Woodland et al., 1995]. The acoustic training data consisted of 36493 sentences from the SI-284 WSJ0 and WSJ1 sets, and the LIMSI 1993 WSJ lexicon and phone set were used. The system was trained using decision-tree-based state-clustering [Young et al., 1994] to define 6399 speech states. A 12 component mixture Gaussian distribution was then trained for each tied state to give a total of about 6 million parameters. The speech data was parameterised into 12 MFCCs, C_1 to C_{12}, along with normalised log-energy and the first and second differentials of these parameters. This yielded a 39-dimensional feature vector to which cepstral mean normalisation was applied. In the GS schemes, a 512 codeword VQ codebook was used.

The data used for evaluation were the ARPA 1994 H1 development and evaluation test sets. The 1994 ARPA H1 task is an unlimited vocabulary task recorded in a clean[9] environment. A 65k word list and dictionary were used with a trigram language model as described in [Woodland et al., 1995]. All decoding used a dynamic-network decoder [Odell et al., 1994] which can either operate in a single-pass or rescore pre-computed word lattices. For efficiency reasons, the results presented here are based on lattice rescores[10]. This means that the percentage of Gaussians calculated is higher than would be the case if a full recognition run was used.

6.2 Results

Table 1 goes here

Initially the performance was assessed by calculating the forced alignment likelihoods using the recognition from the standard system[11]. This allows the measure in equation 10 to be directly calculated. Table 1 shows the results for SGS and SBGS schemes. The top line represents the performance of the standard no GS system. Using SGS there is little degradation in likelihood

[9] Here the term "clean" refers to the training and test conditions being from the same microphone type with a high signal-to-noise ratio.

[10] Large lattices were used in this task, so recognition performance is not expected to be significantly affected.

[11] Strictly the state boundaries positions should be fixed in the alignment task. For the results in this work the state boundaries are allowed to move, so the figures given will be a slight overestimate of the "true" numbers. This is felt to be a second-order affect and is not expected to alter the relative values significantly.

score. This indicates that the use of tail thresholding schemes yields a good choice of Gaussians. Using SBGS reduced the likelihood score.

Table 2 goes here

The figures in table 1 indicate that SBGS removes some of the Gaussians that should be calculated, but gives no indication of how this affects performance. Table 2 shows recognition rate and percentage of Gaussians calculated. As expected the SGS scheme showed little degradation in performance, however 36% of the Gaussians were required to be calculated. Despite the drop in likelihood score, the degradation in recognition performance for the SBGS schemes was small. On the evaluation task using a two ring approach with 5 Gaussian components in the inner ring, calculating only 17.4% of the Gaussians resulted in only a slight degradation of 3.3% in relative performance. When only 4 Gaussian components are allowed in the inner ring, degradations are seen in both the likelihood and recognition performance.

Table 3 goes here

Before presenting recognition results for MLGS and OGS, it is interesting to note the effects that the various occupancy thresholds have on the number of Gaussians selected per context. Table 3 shows the percentage of states assigned to each context group according to the value of l_d and l_i. Naturally as l_d increased the percentage assigned at the context dependent level decreased, this is approximately equivalent to reducing Θ. This of course has an affect on the number of Gaussian components assigned to each codeword. For example, for MLGS with $N_1 = 6, N_2 = 2, N_3 = 1, \Theta = 1.9$, the average number of Gaussian components per codeword decreased from 6189 to 5735 when l_d was increased from 10.0 to 20.0 (from a total of over 75000 Gaussian components).

Table 4 goes here

As before, the performance of the MLGS scheme was first assessed in terms of the likelihood reduction when forced aligning to the recognition transcriptions of the standard system. These are illustrated in table 4. At the low occupancy count of $l_d = 10$ there is little degradation in likelihood for the 12 Gaussian component case. However reducing the number of Gaussian components or increasing l_d does affect the likelihood score. It is interesting to compare the MLGS scheme with the OGS scheme. At the same thresholds there was a smaller reduction in likelihood using OGS than MLGS. This indicates that the simple decision rule used in MLGS is sub-optimal. Unfortunately the OGS scheme is only applicable at the context dependent level prior to any backing-off (MLGS is used to assign the other states here). However, this does indicate that the use of a more complex selection process such as that described in appendix A may be beneficial.

Table 5 goes here

From table 4, reducing the value of N_1 from 6 to 4 seriously affected the likelihood. N_1 was therefore set to 6 and various parameters investigated in terms of the percentage of Gaussians calculated and word error rates. The results are shown in table 5. Comparing these performance figures with those in table 2 shows that similar performance may be obtained at a reduced number of Gaussians calculated. For example on the evaluation data a performance degradation of only around 3% may be achieved with under 14% of the Gaussians calculated, compared to over 17% with SBGS. It is also worth noting the performance when $N_2 = 1$ where the percentage of Gaussians calculated is reduced by around 20%. This reduction is not surprising given that around 50% of the states (see table 3) are assigned to the context independent grouping. Hence, any reduction in the number of Gaussian components taken per state from this group significantly lowers the overall number of Gaussians. However, this is at some cost to the recognition performance, for example the development data results are degraded by around 5% (or 2% relative to the $N_2 = 2$ case).

It is worth pointing out that the MLGS approach requires the setting of more variables in training than either SGS or SBGS. However, MLGS has the advantage that the choice of thresholds may be explicitly related to values measured from the training data. For the other schemes, less explicit methods have to be used.

Table 6 goes here

Finally, table 6 shows the performance of a series of comparable GS schemes (in terms of recognition accuracy). All the state-based Gaussian schemes show significant reductions in the percentage of Gaussians calculated compared to the SGS scheme. The performance of the MLGS and OGS schemes show further improvements compared to the SBGS scheme.

The performance of the various GS schemes may also be compared with directly reducing the number of Gaussians used to model the data associated with each state. Halving the number of speech Gaussians to six per state gave an error rate of 10.08% and 10.91% on the H1 development and evaluation sets respectively. These figures are significantly worse than any of the GS schemes, both in terms of increasing the word error rate and percentage of Gaussians calculated.

The experiments presented in this section illustrate some of the possible options, and their affects on recognition performance, for improved GS. For practical systems other issues besides the percentage of Gaussians computed must be considered in computing the overall reduction in computational load. For example, storing the codeword entries is an additional memory requirement that could possibly increase paging. These considerations could lead to lower overall reductions in computational load. Nonetheless, the importance of incorporating state information in the GS process clearly has large potential gains over standard schemes.

7 Conclusions

This paper has considered the problem of Gaussian selection. In particular it has addressed the problem of, having split the acoustic space into a set of clusters, how to select the "best" shortlist of Gaussians to associate with each cluster. Various new selection schemes have been introduced based on the use of state information, specifically which state each of the Gaussian components belongs to. First a simple extension of standard GS was described, SBGS, where a constraint on the maximum number of Gaussian components per state was applied. Then more complex schemes, MLGS and OGS, were presented based on selecting Gaussians such that the likelihood of the training data is maximized. Techniques for handling limited training data were also described. The techniques using state information were compared with the standard GS scheme on a large vocabulary task, the 1994 ARPA Hub 1. On this task a substantial reduction in the computation of Gaussians was achieved. Despite the fact that lattice rescores were used, thus yielding a higher percentage of Gaussians calculated than if a full system was run, less than 15% of the Gaussians were required to be calculated with little degradation in performance.

In this paper only the simpler versions of maximum likelihood Gaussian selection have been examined. Future work will use decision trees to handle limited training data and more complex selection schemes.

8 Acknowledgments

Mark Gales was supported by a Research Fellowship from Emmanuel College, Cambridge. Kate Knill was funded by Hewlett-Packard Labs, Bristol, UK.

A A More Accurate Algorithm for MLGS

As described in section 5.1 the aim of MLGS is to select the Gaussians for the shortlist associated with state j and cluster χ_ϕ according to

$$\hat{\nu}_\phi = \arg\max_{\mathcal{V}_\phi} \left\{ \sum_{t=1}^{T} \gamma_j(t) p(\chi_\phi | \mathbf{o}(t)) \log \left(\sum_{G(jm)\in\mathcal{V}_\phi} w_{jm} \mathcal{N}(\mathbf{o}(t); \mu_{jm}, \mathbf{\Sigma}_{jm}) \right) \right\} \tag{17}$$

where the maximization is over all shortlists of the appropriate length. It is too expensive, both computationally and in terms of memory requirements, to investigate every possible Gaussian component grouping within a state. However by using a scheme similar to the clustering scheme proposed by Chou [Chou, 1991] a relatively efficient selection scheme may be devised. Two additional assumptions are made. First, all the data from a particular Gaussian component, according to the no GS scheme, is re-assigned in a similar way in the system with GS. Second the frame-state component alignment is not significantly altered. The selection process therefore becomes

$$\hat{\nu}_\phi = \arg\max_{\mathcal{V}_\phi} \left\{ \sum_{G(jm)\in\mathcal{V}_\phi} \sum_{t=1}^{T} \gamma_j(t) \gamma'_m(t) \log \left(w_{jm} \mathcal{N}(\mathbf{o}(t); \mu_{jm}, \mathbf{\Sigma}_{jm}) \right) \right\} \tag{18}$$

where $\gamma'_m(t)$ is the probability that the observation at time t was assigned to Gaussian component m in the hypothesized shortlist, subject to the previously stated constraint that all data from the same Gaussian component is reassigned in a similar way.

The problem is how to assign all the data to one of the Gaussian components in the shortlist, i.e. find $\gamma'_m(t)$. As the data must be re-assigned on a Gaussian component level, $\mathcal{D}_{jm\phi}$ will be used to represent all the data associated with Gaussian component m of state j and cluster χ_ϕ, and $\gamma'_{jm\phi}(k)$ to represent the assignment of all the data from Gaussian component k to Gaussian component m in the shortlist (see below). The selection process in equation 18 may then be rewritten as

$$\hat{\nu}_\phi = \arg\max_{\mathcal{V}_\phi} \left\{ \sum_{G(jm)\in\mathcal{V}_\phi} \sum_{G(jk)} \gamma'_{jm\phi}(k) \left[\mathcal{L}(\mathcal{D}_{jk\phi}; \mu_{jm}, \mathbf{\Sigma}_{jm}) + \sum_{t=1}^{T} L_{jk\phi}(t) \log(w_{jm}) \right] \right\} \tag{19}$$

where

$$\mathcal{L}(\mathcal{D}_{jk\phi}; \mu_{jm}, \mathbf{\Sigma}_{jm}) = -\frac{1}{2} \Bigg[\left(K_{jm} + \mu_{jm}^T \mathbf{\Sigma}_{jm}^{-1} \mu_{jm}\right) \sum_{t=1}^{T} L_{jk\phi}(t) + \\ \mathrm{tr}\left(\mathbf{\Sigma}_{jm}^{-1} \sum_{t=1}^{T} L_{jk\phi}(t) \mathbf{o}(t)\mathbf{o}(t)^T \right) + \mu_{jm}^T \mathbf{\Sigma}_{jm}^{-1} \sum_{t=1}^{T} L_{jk\phi}(t) \mathbf{o}(t) \Bigg] \tag{20}$$

and

$$L_{jk\phi}(t) = \gamma_{jk}(t)p(\chi_\phi|\mathbf{o}(t)) \tag{21}$$

K_{jm} is the standard normalising term associated with Gaussian $G(jm)$.

Two assignment schemes may be used. First a "hard" assignment may be used according to

$$\gamma'_{jm\phi}(k) = \begin{cases} 1, & \text{iff } G(jm) = \arg\max\limits_{G(jm)\in\mathcal{V}_\phi} \left\{\mathcal{L}(\mathcal{D}_{jk\phi};\mu_{jm},\mathbf{\Sigma}_{jm}) + \sum_{t=1}^{T} L_{jk\phi}(t)\log(w_{jm})\right\} \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

An alternative is to use a soft assignment.

$$\gamma'_{jm\phi}(k) = \frac{\exp\left(\mathcal{L}(\mathcal{D}_{jk\phi};\mu_{jm},\mathbf{\Sigma}_{jm}) + \sum_{t=1}^{T} L_{jk\phi}(t)\log(w_{jm})\right)}{\sum_{G(jm)\in\mathcal{V}_\phi}\exp\left(\mathcal{L}(\mathcal{D}_{jk\phi};\mu_{jm},\mathbf{\Sigma}_{jm}) + \sum_{t=1}^{T} L_{jk\phi}(t)\log(w_{jm})\right)} \tag{23}$$

This is analogous to Baum-Welch re-estimation in contrast to Viterbi re-estimation. Both these schemes re-assign data in an optimal ML sense.

B Handling Limited Training Data with Decision Trees

There are a variety of standard schemes to ensure that HMM parameters are robustly estimated. Many of these schemes may be applied to ensure robust shortlist generation for GS.

One currently popular scheme for robustly estimating parameters in speech recognition is to use a decision tree approach [Bahl et al., 1991, Young et al., 1994]. This allows a smooth sharing of data between the various contexts of a particular phone, or even between phones. Assuming that the speech recogniser has been built using decision trees[12], the same trees may be used in the Gaussian selection process. To ensure the selection process is robust a minimum occupancy threshold is set. A simple top-down scheme starts the search at the root node (assuming that the root node has sufficient data) for each cluster and then:

For a node A

if A has children B and C

if $\sum_{Leaf_B}\sum_{t=1}^{T}\gamma_j(t)p(\chi_\phi|\mathbf{o}(t)) > Thresh$

search through node B

else

let all Gaussian components of subtree A be associated with all the states of subtree B

if $\sum_{Leaf_C}\sum_{t=1}^{T}\gamma_j(t)p(\chi_\phi|\mathbf{o}(t)) > Thresh$

search through node C

[12]If this is not the case a separate decision tree may be built.

else

let all Gaussian components of subtree A be associated with all the states of subtree C

else

let all Gaussian components of subtree A be associated with all the states of subtree A

where Leaf_B indicates all leaves of subtree B. Using this scheme the set of Gaussian components to be associated with each state is easily derived. The selection process to be used determines how this information is to be used. For the standard MLGS scheme the following selection process is used.

$$G(jm) \in \nu_\phi \;\text{ iff }\; G(jm) \in \arg\max_m(n) \left\{ \sum_{k \in \mathbf{s}(j)} \sum_{t=1}^{T} \gamma_k(t) p(\chi_\phi|\mathbf{o}(t)) \log\left(w_{jm}\mathcal{N}(\mathbf{o}(t);\mu_{jm},\mathbf{\Sigma}_{jm})\right) \right\} \tag{24}$$

where $\mathbf{s}(j)$ is the list of all Gaussian components associated with state j as determined by the decision tree. For the accurate MLGS scheme described in appendix A, all Gaussian components associated with state j must be assigned to one of the Gaussian components in the hypothesised shortlist, i.e. $\gamma'_{jm\phi}(k)$ is calculated for all Gaussian components.

References

[Bahl et al., 1991] Bahl, L. R., de Souza, P. V., Gopalkrishnan, P. S., Nahamoo, D., and Picheny, M. A. (1991). Context dependent modelling of phones in continuous speech using decision trees. In *Proceedings DARPA Speech and Natural Language Processing Workshop*, pages 264–270.

[Beyerlein, 1994] Beyerlein, P. (1994). Fast log-likelihood computation for mixture densities in a high-dimensional feature space. In *Proc. ICSLP*, pages 271–274, Yokohama.

[Beyerlein and Ullrich, 1995] Beyerlein, P. and Ullrich, M. (1995). Hamming distance approximation for a fast log-likelihood computation for mixture densities. In *Proc. Eurospeech*, pages 1083–1086, Madrid.

[Bocchieri, 1993] Bocchieri, E. (1993). Vector quantization for efficient computation of continuous density likelihoods. In *Proc. ICASSP*, volume II, pages II–692–II–695, Minneapolis.

[Chou, 1991] Chou, P. (1991). Optimal partitioning for classification and regression trees. *IEEE Trans PAMI*, 13:348.

[Fritsch and Rogina, 1996] Fritsch, J. and Rogina, I. (1996). The bucket box intersection (BBI) algorithm for fast approximate evaluation of diagonal mixture gaussians. In *Proc. ICASSP*, volume 2, pages II–273–II–276, Atlanta.

[Huang et al., 1990] Huang, X. D., Lee, K. F., and Hon, H. W. (1990). On semi-continuous hidden Markov modelling. In *Proc. ICASSP*, pages 689–692.

[Hunt and Lefebre, 1989] Hunt, M. and Lefebre, C. (1989). A comparison of several acoustic representations for speech recognition with degraded and undegraded speech. In *Proceedings ICASSP*, pages 262–265.

[Knill et al., 1996] Knill, K. M., Gales, M. J. F., and Young, S. (1996). Use of Gaussian selection in large vocabulary continuous speech recognition using HMMs. In *Proc. ICSLP*, volume 1, pages I–470–I–473, Philadelphia.

[Linde et al., 1980] Linde, Y., Buzo, A., and Gray, R. M. (1980). An algorithm for vector quantizer design. *IEEE Trans Comms*, COM-28(1):84–95.

[Murveit et al., 1994] Murveit, H., Monaco, P., Digalakis, V., and Butzberger, J. (1994). Techniques to achieve an accurate real-time large-vocabulary speech recognition system. In *Proc. ARPA Workshop on Human Language Technology*, pages 368–373, Plainsboro, N.J.

[Odell et al., 1994] Odell, J. J., Valtchev, V., Woodland, P. C., and Young, S. J. (1994). A one pass decoder design for large vocabulary recognition. In *Proceedings ARPA Workshop on Human Language Technology*, pages 405–410.

[Ortmanns et al., 1997] Ortmanns, S., Firzlaff, T., and Ney, H. (1997). Fast likelihood computation methods for continuous mixture densities in large vocabulary speech recognition. In *Proceedings Eurospeech*, pages 139–142.

[Pallett et al., 1995] Pallett, D. S., Fiscus, J. G., Fisher, W. M., Garofolo, J. S., Lund, B. A., Martin, A., and Przybocki, M. A. (1995). 1994 benchmark tests for the ARPA spoken language program. In *Proceedings ARPA Workshop on Spoken Language Systems Technology*, pages 5–36.

[Poggi, 1993] Poggi, G. (1993). Fast algorithm for full-search VQ encoding. *Electronics Letters*, 29(12):1141–1142.

[Woodland et al., 1995] Woodland, P. C., Odell, J. J., Valtchev, V., and Young, S. J. (1995). The development of the 1994 HTK large vocabulary speech recognition system. In *Proceedings ARPA Workshop on Spoken Language Systems Technology*, pages 104–109.

[Young et al., 1994] Young, S. J., Odell, J. J., and Woodland, P. C. (1994). Tree-based state tying for high accuracy acoustic modelling. In *Proceedings ARPA Workshop on Human Language Technology*, pages 307–312.

C Biographies

Mark J.F. Gales studied for the B.A. in Electrical and Information Sciences at the University of Cambridge from 1985-88. Following graduation he worked as a consultant at Roke Manor Research Ltd. In 1991 he took up a position as a Research Associate in the Speech Vision and Robotics group if the Engineering Department at Cambridge University. In 1996 he completed his doctoral thesis: Model-Based Techniques for Robust Speech Recognition. From 1995-1997 he was a Research Fellow at Emmanuel College Cambridge. He is currently a Research Staff Member in the Speech goup at the IBM T.J.Watson Research Center. His research interests include robust speech recognition, speaker adaptation, segmental models of speech and large-vocabulary speech recognition.

Katherine M. Knill (S'90, M'94) received the BEng Degree in electronic engineering and mathematics from Nottingham University, England, in 1990, and the PhD and DIC degrees from Imperial College, London University, England, in 1994. She was sponsored during this time (1986-1993) by Marconi Underwater Systems Ltd, England. In October 1993 she joined the Speech, Vision and Robotics Group at Cambridge University, England. Her primary research focus was keyword spotting for audio document retrieval, in collaboration with Hewlett-Packard Labs, Bristol, England. She is currently working for Nuance Communications, Menlo Park, CA, which she joined in January 1997. A member of the algorithms group, she works on developing commercial telephone-based automatic speech recognition system.

Stephen J. Young completed the PhD in speech synthesis in 1977 after taking a First in electrical sciences, both at Cambridge University, UK. He took a Lectureship at UMIST before joining the Cambridge University Engineering Department in 1984 as a Lecturer and Fellow of Emmanuel College. He was promoted to a Readership in October 1994 and elected to the Chair of Information Engineering in December 1994. His research intersets include speech recognition, language modelling and multimedia applications. He is currently head of the Speech, Vision and Robotics (SVR) Group and he is Technical Director of Entropic Cambridge Research Ltd . He is a Fellow of the Institute of Acoustics and European Editor of Computer Speech and Language.

Selection scheme	Thresholds				Average Likelihood
	Θ_1	N_1	Θ_2	N_2	(change per frame)
—	—	—	—	—	0.000
SGS	1.9	—	—	—	-0.021
SBGS	1.3	5	1.9	1	-0.826
	1.3	4	1.9	1	-1.125

Table 1: Change in the average forced alignment likelihood of the ARPA 1994 H1 development data for SGS and SBGS systems, compared to the standard no GS system.

Selection scheme	Thresholds				H1 Dev		H1 Eval	
	Θ_1	N_1	Θ_2	N_2	%Gauss	%Error	%Gauss	%Error
—	—	—	—	—	100.0	9.57	100.0	9.20
SGS	1.9	—	—	—	36.0	9.72	35.6	9.27
SBGS	1.3	5	1.9	1	17.6	9.79	17.4	9.58
	1.3	4	1.9	1	15.9	9.95	15.8	9.85

Table 2: Recognition performance of the standard no GS, SGS and SBGS systems on the ARPA 1994 H1 task.

Codewords	Thresh.		%States Assign.		
Φ	l_d	l_i	$\lambda^{(d)}$	$\lambda^{(i)}$	$\lambda^{(\phi)}$
512	10.0	50.0	11.9	45.0	43.1
	20.0	50.0	8.2	48.4	43.4
	50.0	50.0	4.6	51.9	43.5

Table 3: The percentage of states assigned at each context level in MLGS and OGS. $\lambda^{(d)}$ is the context dependent level, $\lambda^{(i)}$ is the context independent level and $\lambda^{(\phi)}$ is cluster level, with $\Theta = 1.9$.

Selection scheme	Max. Cmpts.			Average Likelihood
	N_1	N_2	N_3	(change per frame)
MLGS ($l_d = 10$)	12	2	1	-0.241
	6	2	1	-0.621
	4	2	1	-1.048
MLGS ($l_d = 20$)	6	2	1	-0.660
	6	1	1	-0.785
MLGS ($l_d = 50$)	6	2	1	-0.746
OGS ($l_d = 20$)	6	2	1	-0.555

Table 4: Change in the average forced alignment likelihood of the ARPA 1994 H1 development data for MLGS and OGS with $l_i = 50$, compared to the standard no GS system.

Selection scheme	Thresh.		H1 Dev		H1 Eval	
	l_d	l_i	%Gauss	%Error	%Gauss	%Error
MLGS ($N_2 = 2$)	10.0	50.0	15.4	9.78	15.2	9.35
	20.0	50.0	14.1	9.84	13.9	9.46
	50.0	50.0	12.6	10.11	12.5	9.41
MLGS ($N_2 = 1$)	20.0	50.0	11.4	10.02	11.2	9.49
OGS ($N_2 = 2$)	20.0	50.0	14.1	9.78	13.9	9.56

Table 5: Recognition performance of MLGS and OGS on the ARPA 1994 H1 task with $N_1 = 6$, $N_3 = 1$ and $\Theta = 1.9$.

Selection scheme	H1 Dev		H1 Eval	
	%Gauss	%Error	%Gauss	%Error
—	100.0	9.57	100.0	9.20
SGS	36.0	9.72	35.6	9.27
SBGS	17.6	9.79	17.4	9.58
MLGS	14.1	9.84	13.9	9.46
OGS	14.1	9.78	13.9	9.56

Table 6: Summary of comparative performance for the various selection schemes on the ARPA 1994 H1 Task.