# USE OF GAUSSIAN SELECTION IN LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION USING HMMS

*K.M.Knill, M.J.F.Gales and S.J.Young*

Cambridge University Engineering Department,
Cambridge, CB2 1PZ, UK
{kmk,mjfg,sjy}@eng.cam.ac.uk

## ABSTRACT

This paper investigates the use of Gaussian Selection (GS) to reduce the state likelihood computation in HMM-based systems. These likelihood calculations contribute significantly (30 to 70%) to the computational load. Previously, it has been reported that when GS is used on large systems the recognition accuracy tends to degrade above a $\times 3$ reduction in likelihood computation. To explain this degradation, this paper investigates the trade-offs necessary between achieving good state likelihoods and low computation. In addition, the problem of unseen states in a cluster is examined. It is shown that further improvements are possible. For example, using a different assignment measure, with a constraint on the number of components per state per cluster, enabled the recognition accuracy on a 5k speaker-independent task to be maintained up to a $\times 5$ reduction in likelihood computation.

## 1. INTRODUCTION

In recent years, high accuracy large vocabulary continuous speech recognition systems have been developed. However, most systems, in particular those based on HMMs, have tended to operate at several times real-time. To convert laboratory systems into useful products, techniques are required to reduce the decoding time to faster than real-time, while maintaining, or staying close to, the same level of accuracy. A number of methods have been proposed in the literature, falling into the following general categories; pruning, tying, feature selection, and Gaussian selection. Since the evaluation of Gaussian likelihoods can dominate the total computational load, taking between 30 to 70% of the computation, the latter category is of particular interest and is the topic of this paper.

The motivation behind Gaussian Selection (GS) is as follows. If an input vector is an outlier with respect to a component distribution, i.e. it lies on the tail of the distribution, then the likelihood of that component producing the input vector is very small. This results in the component likelihoods within a state having a large dynamic range, with one or two components tending to dominate the state likelihood for a particular input vector. Hence, the state likelihood could be computed solely from these components without a noticeable loss in accuracy. GS methods attempt to efficiently select these components, or a subset containing them, at each frame.

GS was originally proposed by Bocchieri [3]. The acoustic space is divided up during training into a set of vector quantised regions. Each component is then assigned to one or more VQ codewords. During recognition, the input speech vector is vector quantised. Only the likelihoods of the components associated with the VQ codeword corresponding to the observation vector are computed exactly and the remaining likelihoods are approximated. In Bocchieri's work, the VQ codebooks were generated by clustering the Gaussian components. An alternative data driven approach was proposed by Murveit et al [7]. Recently a number of direct search methods have been proposed. However, these are typically system specific, e.g. requiring a single shared covariance [1], or a large number of components per state [4].

This paper investigates the GS approach proposed by Bocchieri. In particular, the paper addresses the problem of limited performance when pruning (beam-search) is applied. Bocchieri reported that on context-independent systems with no pruning reductions of up to $\times 9$ in the likelihood computation can be made [3]. However, this drops to $\times 3$ when pruning is applied in a context-dependent system [2]. The causes of this degradation are investigated on the CUED-HTK system used on the 5k Hub 2 task in the 1993 ARPA evaluation [9], reimplemented in HTK V2.0 [10].

## 2. GAUSSIAN SELECTION METHOD

The Gaussian clustering was performed during HMM training as follows. A weighted (Mahalanobis-like) Euclidean distance between the means was used to calculate the distance between the $i$th and $j$th Gaussians.

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

where $D$ is the feature vector dimension, $\mu_i(k)$ is the $k$th component of vector $\mu_i$, and $w(k)$ is equal to the inverse square root of the $k$th 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 $m$th Gaussian component. Then, a clustering procedure based on the Linde-Buzo-Gray [6] algorithm was applied to minimise the average (per Gaussian component) distortion, $\delta_{avg}$,

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

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

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

The clusters produced by the above process are disjoint. If used in recognition, errors are likely since the likelihood of some Gaussians close to the input feature vector but not in the selected cluster will be excluded from the full computation. To avoid this, clusters share Gaussians as follows. Given a threshold $\Theta \gg 1$, an input feature vector, $\mathbf{o}$, is said to fall on the tail of the $m$th Gaussian if

$$\frac{1}{D} \sum_{i=1}^{D} \frac{(o(i) - \mu_m(i))^2}{\sigma_m^2(i)} > \Theta \quad (4)$$

where $\sigma_m^2(i)$ is the $i$th diagonal element of $\mathbf{\Sigma}_m$. Thus, if the cluster centroid is taken to be a typical input feature vector the neighbourhood, $\nu_\phi$, of codeword $\mathbf{c}_\phi$, can be defined as consisting of all Gaussians such that [3]

$$G(\mu_m, \mathbf{\Sigma}_m) \in \nu_\phi \;\; iff \;\; \frac{1}{D} \sum_{i=1}^{D} \frac{(c_\phi(i) - \mu_m(i))^2}{\sigma_{avg}^2(i)} \leq \Theta \quad (5)$$

where $\sigma_{avg}^2(i)$ is the $i$th diagonal element of the matrix of the average covariance of the full Gaussian set. The use of $\sigma_{avg}^2(i)$ in the criterion is preferred to $\sigma_m^2(i)$ since the individual variance estimates are often noisy.

The *weighted* distance measure in equation 5 takes no account of the variance of the cluster, so some clusters grow too large. An alternative, *class-weighted*, distance measure was therefore implemented, where the neighbourhood of codeword, $\mathbf{c}_\phi$, consists of the Gaussians such that

$$G(\mu_m, \mathbf{\Sigma}_m) \in \nu_\phi \;\; iff \;\; \frac{1}{D} \sum_{i=1}^{D} \frac{(c_\phi(i) - \mu_m(i))^2}{\sqrt{(\sigma_{avg}^2(i)\sigma_{c_\phi}^2(i))}} \leq \Theta \quad (6)$$

where $\sigma_{c_\phi}^2(i)$ is the $i$th diagonal element of the cluster centroid covariance, $\mathbf{\Sigma}_{\mathbf{c}_\phi} = 1/size(\chi_\phi) \sum_{m \in \chi_\phi} \mathbf{\Sigma}_m$.

During recognition, the appropriate neighbourhood is selected by determining the cluster centroid, $\mathbf{c}_i$, which minimises the weighted (Mahalanobis-like) Euclidean distance to the observation vector, $\mathbf{o}(t)$, at time $t$,

$$\mathbf{c}_i = \min_{\phi=1}^{\Phi} \delta(\mathbf{o}(t), \mathbf{c}_\phi) \quad (7)$$

Each state likelihood is calculated by exactly determining the log likelihoods of components within the selected cluster and approximating, by a discrete value, the other components. The reduction in computation of the Gaussians, the computation fraction, $C$, is defined as

$$C = \frac{G_{new} + VQ_{comp}}{G_{full}} \quad (8)$$

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 choice of $\Theta$ in equation 6 controls the average size of the Gaussian neighbourhoods. Efficiency improves with reductions in $\Theta$ 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 component assigned to a selected cluster, errors can occur if some components that make a significant contribution to a state likelihood are not contained in that cluster. The state likelihood is likely to be poorly approximated in this case. Further errors can occur due to 'state flooring'. When no components from a state are assigned to the selected cluster the state likelihood is simply given a discrete approximate value, i.e. it is floored. Since it is possible for an input vector to be an outlier with respect to all the component distributions of a state on the optimal path, the state flooring chosen can be crucial to maintaining accuracy. This is investigated in the next section.

A good cluster assignment is therefore one which assigns all, or most of, the components that contribute significantly to state likelihoods for that region of acoustic space, while assigning as few non-contributing components as possible.

## 3. EXPERIMENTAL RESULTS

### 3.1. Experimental Setup

The November 1993 ARPA WSJ evaluation Hub 2 task consists of a 5k word closed vocabulary, with a standard bigram language model provided by Lincoln Labs, the 5K Closed NVP bigram. Pronunciations were taken from the Dragon Wall Street Journal Pronunciation Lexicon version 2.0 with some locally generated additions and corrections.

The experiments reported here used the HMM model set trained for the Hub 2 task. This set consists of continuous mixture density, 8 mixture component, tied state, word-internal triphone HMMs corresponding to a 44 base phone set plus silence and optional inter-word silence models. All the speech phone models had three emitting states, and a strictly left-to-right topology. Acoustic features are 12 MFCCs and log energy plus first and second derivatives (total 39 dimensions). State tying was performed using a decision-tree based algorithm. More details may be found in [9]. The decoding was performed using the HTK V2.0 tool HVite [10].

HTK V2.0 [10] was also used to build the GS codebooks. The codebooks were flat with 256 codewords. A standard flat search was implemented to eliminate effects of codebook search errors from the results (faster methods such as [8] can reduce the search cost to a few percent relative to the standard full search). Results are given for both lattice rescoring and full recognition experiments. The former were performed for speed of execution, and serve to illustrate the general behaviour. Since the codebook search is disproportionate to the search-space of the lattices, reductions in computation are only considered in terms of the percentage of component likelihoods computed with GS with respect to the none GS lattice system. These percentages tend to be slightly high with respect to the actual reductions that can be achieved in the full recognition system. For this full system, computation reduction is presented using the computation fraction defined in equation 8.

## 3.2. Gaussian To Cluster Assignment

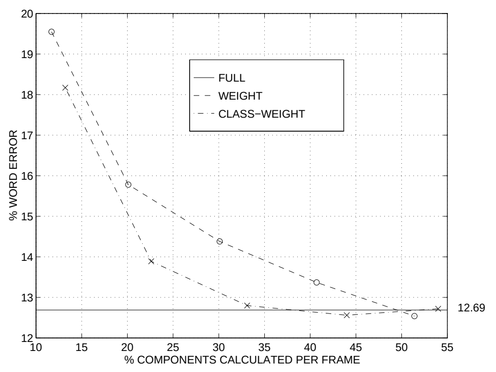

**Figure 1:** Word error against % component likelihoods calculated per frame for weighted and class-weighted assignment measures at tail thresholds {1.0,1.3,1.6,1.9,2.2} using lattice rescoring.

It was previously reported that a cluster class-based ("*class-weight*") distance measure improved the assignment of components to cluster neighbourhoods compared to a standard average weighted distance measure on a SD word-spotting task [5]. Figure 1 shows that this holds for a speaker-independent LVCSR task. All further results are, therefore, presented using the class-weighted measure.

## 3.3. State Flooring

From figure 1, it can be seen that the class-weighted system accuracy matches the non-GS system for tail thresholds of 1.6 and above. Even at these thresholds the choice of approximate log likelihood for components outside the frame cluster was found to be important. If the likelihood is too small reasonable paths may be killed, whereas poor paths may remain within the search space if the likelihood is too large. The choice of approximation is especially sensitive when no components from a state belong to the selected cluster so 'state flooring' occurs. A good approximate log likelihood score was found empirically.

To test whether the GS performance degrades below the 1.6 tail threshold due to the state likelihoods of components within a selected cluster being poorly determined or due to state flooring, the 1.3 system was run as follows. If no components from a state belonged to the VQ codeword at a particular frame then the true likelihood of that state was computed. Otherwise, the state likelihood was computed only from those components lying within the codeword cluster. With this approach, the 1.3 system was able to attain the standard system accuracy. This indicates that a significant part of the failing of the lower thresholds is due to state flooring.

As a very basic solution, each cluster was forced to contain at least one component from each state. If no components from a state met the tail threshold, then the closest component to the cluster centroid from that state was assigned to the cluster. Since no states were floored no approximations were required for the components outside the cluster. Under this constraint, word error rates close to the standard system (12.69%) were observed throughout the tail threshold range, but at the cost of an increase in the number of components calculated, as shown in table 1.

| Tail | Standard GS | | Min 1 CSC | |
|---|---|---|---|---|
| | % Lhood calc | Word error | % Lhood calc | Word error |
| 1.0 | 13.17 | 18.17 | 23.37 | 13.07 |
| 1.3 | 22.55 | 13.89 | 30.31 | 12.88 |
| 1.6 | 33.08 | 12.80 | 38.77 | 12.72 |

**Table 1:** Word error against % component likelihoods calculated per frame for the standard GS system and for GS with at least one component per state per codeword (CSC) at assignment tail thresholds of {1.0,1.3,1.6} using lattice rescoring.

## 3.4. Constrained Gaussian Selection

The good performance at tail thresholds of 1.6 and above is achieved at the expense of excess computation. Examination of the cluster sets at these thresholds showed that many clusters contained several components from the same state. This is in contradiction with the motivation behind GS, namely that the state likelihood calculation is dominated by one or two components, implying that some unnecessary computation is being performed. As a first, simple, solution to this problem the number of CSC was limited to a fixed number, $S$. If more than $S$ components from a state satisfied the tail threshold then the closest $S$ components to the codeword centroid were assigned to that cluster.

In addition, the results in table 1 suggest that most of the dominant likelihoods required for exact computation in a cluster lie within the 1.3 tail threshold, with the remaining important components in the region from a tail of 1.3 to 1.9. Thus, a dual ring CSC constraint was applied. In this, the number of CSC was constrained at an inner tail threshold, as in the single ring approach, and between the inner tail and an outer tail threshold. Approximating the components outside the cluster with a discrete log likelihood (since some states were floored), the results shown in table 2 were obtained.

| Approach | Inner tail | Max CSC | Outer tail | Max CSC | Comp Fr C | Word error |
|---|---|---|---|---|---|---|
| Standard | - | - | - | - | 100.0 | 12.72 |
| Standard GS | 1.6 | - | - | - | 24.1 | 12.75 |
| Single ring | 1.6 | 4 | - | - | 18.4 | 12.88 |
| | 1.9 | 3 | - | - | 20.3 | 12.88 |
| | 1.9 | 4 | - | - | 24.3 | 12.46 |
| Double ring | 1.3 | 4 | 1.6 | 1 | 17.7 | 12.69 |
| | 1.3 | 4 | 1.9 | 1 | 20.4 | 12.54 |
| Min 1 CSC | 1.3 | - | - | - | 24.9 | 12.94 |

**Table 2:** Word error against computation fraction for full recognition pass using: standard system, standard class-weight, and class-weighted system with one or two ring constraints on the maximum number of CSC, and with a minimum of 1 CSC.

Table 2 shows that a $\times 4$ reduction in computation was achieved using the standard class-weighted measure without loss in accuracy. Further reductions were achieved without loss in accuracy by applying a single maximum constraint of 4 CSC at a tail of 1.6 or 3 CSC at a tail of 1.9. Tighter constraints led to a drop in performance due to the sub-optimal selection of the components. For example, input vectors at the centre and edge of a cluster are likely to be best fitted to different components within the cluster[1].

The double ring constrained system produced the greatest computation reduction. From table 2 it can be seen that most of the dominant likelihoods lie in the 1.3 tail boundary. In experiments performed with an inner ring tail threshold of 1.0, the accuracy degraded as important components were eliminated from the clusters. As shown by table 1, the 1.3 tail threshold requires some further components from outside this boundary. Table 2 shows that a single component in the outer ring from a tail of 1.3 to 1.6 is sufficient to match the standard system accuracy with a computation fraction of just 17.7%. Hence, a $> 5\times$ reduction in likelihood computation can be achieved without an increase in the word error on a pruned system. The overall reduction in decoding time will depend on the ratio of search to likelihood computation time. This is dependent on the amount of pruning and level of tying used.

## 3.5. Comparison With Pruning

The number of likelihood computations can also be reduced by tightening the pruning. Since it reduces the search space, pruning can have a greater effect on overall computation than GS. In the standard system above a pruning threshold of 300 was used. It was found to be possible to tighten this threshold to 250 before significant degradation in error was noticed. Table 3 shows that at this threshold the decoder requires the same overall amount of computation as the best GS system at a beam-search threshold of 300. However, GS can be applied to the 250 decoder yielding more computation reductions.

| Beam-search threshold | Approach | Comp Fr C | Total Comp wrt 300 | Word error |
|---|---|---|---|---|
| 300 | No GS | 100.0 | 100 | 12.72 |
| | 2-ring GS | 17.7 | 71 | 12.69 |
| 250 | No GS | 100.0 | 70 | 12.95 |
| | 2-ring GS | 20.3 | 52 | 12.93 |

**Table 3:** Word error against computation for full recognition pass with pruning thresholds, 250 and 300, using no GS and the best two-ring constrained GS.

## 4. CONCLUSIONS

For a given input vector, a state likelihood can be computed from a subset of the components belonging to that state. The aim of GS is to pre-select these subsets so that the state likelihoods are determined with as few calculations as possible without degrading the recognition accuracy. This paper has shown that GS introduces errors due to (i) the omission of significant, in terms of their contribution to a state likelihood, components from clusters, and (ii) state flooring. In minimising the errors' effects, the Gaussian clusters become over-assigned, with many clusters containing multiple components from the same state. The resulting extra calculations limit the computation reductions that can be achieved to $\sim \times 3$ when efficient pruning techniques are applied to a standard GS LVCSR.

To reduce the over-assignment of components to clusters, the assignment was first improved by use of a class-weighted distance measure. This achieved a $\times 4$ reduction in computation without loss in accuracy. To further reduce the over-assignment, while retaining the significant components, the number of components per state per cluster (CSC) was constrained. The selection of components was very crude, with those closest to the centroid being assigned. Even so, this constraint enabled computation to be reduced to $\sim \times 5$ with no degradation. In particular, a dual ring approach, in which the number of CSC was constrained at both an inner tail threshold, and between the inner tail and an outer tail threshold, yielded a word error of 12.69% at a computation fraction of 17.7%, compared to 12.72% word error in the system without GS. This was based on the observations that most significant components lie within a fairly tight tail threshold, but inclusion of some components from outside this threshold is essential to prevent state flooring problems.

[1]This approach did not work on a SD word-spotting system as the more tightly clustered (in acoustic space) components were more susceptible to incorrect elimination of a component.

## 5. ACKNOWLEDGEMENTS

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

## 6. REFERENCES

1. P. Beyerlein and M. Ullrich. Hamming distance approximation for a fast log-likelihood computation for mixture densities. In *Proc. Eurospeech*, pages 1083–1086, Madrid, 1995.
2. E. Bocchieri. A study of the beam-search algorithm for large vocabulary continuous speech recognition and methods for improved efficiency. In *Proc. Eurospeech*, volume 3, pages 1521–1524, Berlin, 1993.
3. E. Bocchieri. Vector quantization for efficient computation of continuous density likelihoods. In *Proc. ICASSP*, volume II, pages II–692–II–695, Minneapolis, 1993.
4. J. Fritsch and I. Rogina. 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, 1996.
5. K. M. Knill and S.J. Young. Fast implementation methods for Viterbi-based word-spotting. In *Proc. ICASSP*, volume 1, pages I–522–I–525, Atlanta, 1996.
6. Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Trans Comms*, COM-28(1):84–95, Jan 1980.
7. H. Murveit, P. Monaco, V. Digalakis, and J. Butzberger. 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., Mar 1994.
8. G. Poggi. Fast algorithm for full-search VQ encoding. *Electronics Letters*, 29(12):1141–1142, June 1993.
9. P.C. Woodland, J.J. Odell, and S.J. Young. Large vocabulary continuous speech recognition using htk. In *Proc. ICASSP*, volume II, pages 125–128, Adelaide, 1994.
10. S. Young, J. Jansen, J. Odell, D. Ollason, and P. Woodland. *The HTK Book (for HTK V2.0).* Entropic Cambridge Research Laboratory Ltd, Cambridge, U.K., 1996.