Efficient codebook for fast and accurate low resource ASR systems

Leila Zouari, Gérard Chollet

GET - ENST / CNRS-LTCI
Département Traitement du Signal et des Images,
46 rue Barrault, 75314 Paris, France.

Résumé

Nowadays, speech interfaces have become widely employed in mobile devices, thus recognition speed and power consumption are becoming new metrics of Automatic Speech Recognition (ASR) performance.

For ASR systems using continuous Hidden Markov Models (HMMs), the computation of the state likelihood is one of the most time consuming parts. Hence, we propose in this paper novel multi-level Gaussian selection techniques to reduce the cost of state likelihood computation. The proposed algorithms are evaluated within the framework of a large vocabulary continuous speech recognition task.

Key words: Speech, recognition, Gaussian, selection, codebook

1 Introduction

The proliferation of mobile devices in daily life has created a great demand for efficient and simple interfaces. In particular, speech recognition being a key element of the conversational interface, there is a significant requirement for low-resource and accurate automatic speech recognition systems.
Recent mobile devices (GPS [1], GSM [2], PDA [3], ...) offer a large set of functionalities but their resources are too limited for accurate continuous speech recognition engines. Indeed, state-of-the-art continuous speech recognition systems use hidden Markov models (HMM) with many tens of thousands of Gaussian distributions to achieve improved recognition. As the performance and the speed of such systems are closely related to the number of HMM Gaussians,

[1] GPS : Global Positioning System
[2] GSM : Global System for Mobile Communications
[3] PDA : Personal Digital Assistant

reducing the number of Gaussians without decreasing the system performance is of major interest.
According to previous studies (Bocchieri (1993); Mak et al. (2001); Jurgen et al. (1996); Gales et al. (1996)) only a few Gaussians dominate the state likelihood computation. Hence, different techniques were developed to select them.
These methods can be divided into two categories :

– *State based methods :* These methods are often applied to systems having acoustic models with a high number of Gaussians per state such as semi-continuous or monophone based systems (Jurgen et al. (1995); Woszczyna (1998)).

– *Model based methods :* Gaussian selection is applied to the distributions belonging to all the states of the acoustic models. This is for example the case of triphone based recognition systems (Bocchieri (1993); Mak et al. (2001); Gales et al. (1999)).

In this paper, we propose several Gaussian selection techniques that reduce the cost of likelihood computation either in the state or in the model level. The proposed algorithms are evaluated within the framework of a large vocabulary continuous speech recognition task.

2 State-based clustering and selection

The state-based Gaussian selection is performed in two steps : classification and selection. During the first step, state Gaussians are grouped into clusters. Generally, they are organised into a tree structure based on their mean vector values (Ortmanns et al. (1998); Jurgen et al. (1996); Woszczyna (1998)) and the Euclidian distance. The second step consists in selecting Gaussians to be used for the likelihood computation.
In the present section, we introduce a Gaussian similarity metric and empirical criteria to improve the classification process. Then, we evaluate this clustering/classification within the framework of model shortening by vector quantification. Finally, we investigate a state-based multi-level Gaussian selection.

2.1 Gaussian classification

For each Gaussian mixture, distributions are grouped into a binary tree structure and every cut in the tree defines a possible classification. To determine the optimal cut of the tree, two criteria are considered : data driven and dissimilarity based.

2.1.1 Clustering process

The bottom-up clustering algorithm is applied to each mixture of components as follows :

– Compute distances between all pairs of distributions.

– Merge the closest distributions : Let $g_1(n_1, \mu_1, \Sigma_1)$ and $g_2(n_2, \mu_2, \Sigma_2)$ be two Gaussians to which n_1 and n_2 frames have been assigned during the training. If g_1 and g_2 are merged into $g_3(n_3, \mu_3, \Sigma_3)$ then :

$$n_3 = n_1 + n_2 \tag{1}$$

$$\mu_3 = \frac{n_1}{n_1 + n_2}\mu_1 + \frac{n_2}{n_1 + n_2}\mu_2 \tag{2}$$

$$\Sigma_3 = \frac{n_1}{n_1 + n_2}\Sigma_1 + \frac{n_2}{n_1 + n_2}\Sigma_2 + \frac{n_1 n_2}{(n_1 + n_2)^2}(\mu_1 - \mu_2)(\mu_1 - \mu_2)^T \tag{3}$$

g_3 replaces g_1 and g_2 in the set which size is reduced by one.

– If the number of Gaussians is greater than 1 go to the first step.

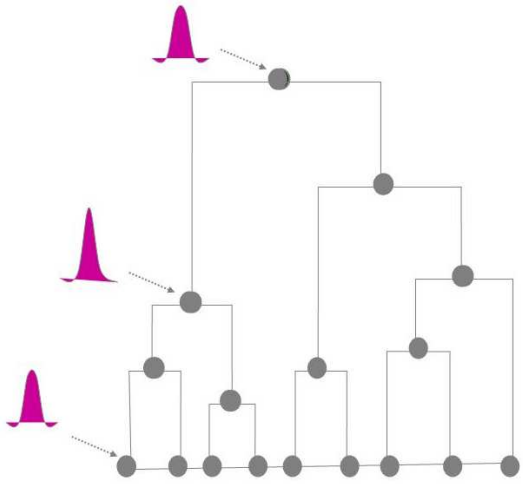


FIG. 1. Hierarchical clustering of Gaussian distributions.

2.1.2 Metrics

Two distances are used : the existing likelihood loss based distance and the proposed weighted relative entropy based metric.

- *Loss likelihood based metric* : If g_1 and g_2 are merged into g_3 then the likelihood loss (PV) is the difference between the likelihoods of g_1 and g_2 and the

likelihood of g_3 :

$$PV(g_1, g_2, g_3) = log \frac{\|\Sigma_3\|^{(n_1+n_2)/2}}{\|\Sigma_1\|^{n_1/2}\|\Sigma_2\|^{n_2/2}} \quad (4)$$

This metric is somewhat similar to the loss of entropy based distance used by Digalakis (Digalakis et al. (2000)). It was successfully used in model adaptation (Mokbel (2001)).

- *The Weighted symmetric Kullback-Leibler divergence (KLP)* : it is expressed as the distance between two probability density functions weighted by the amount of training data.

$$KLP(g_1; g_2) = \frac{1}{2} tr(n_1 \frac{\Sigma_1}{\Sigma_2} + n_2 \frac{\Sigma_2}{\Sigma_1}) + \frac{1}{2}(\mu_1 - \mu_2)^T (\frac{n_1}{\Sigma_1} + \frac{n_2}{\Sigma_2})(\mu_1 - \mu_2) - (n_1 + n_2)d \quad (5)$$

where d is the dimension of the parameters vectors.

The information provided by the amount of training data is only advantageous if training and testing data have the same proportions.

2.1.3 Tree cutting

From the root of the tree to the leaves, cuts result in many different classifications. Three cutting ways are proposed :

– *Fixed* : We consider a constant number of classes. So, the tree is traversed from the leaves till the number of nodes reaches the predefined number of classes.

– *Weight based* : The number of classes depends on the amount of training data. So the tree is processed (from the root) and we stop at the node whose children's weight is less than a predefined threshold.

– *Distance based* : The tree cutting is performed when the distance between two levels reaches a maximum value.

For weight and distance criteria, the number of Gaussians per state is variable. Hence, a mean value is computed.

Codewords refer to the nodes (Gaussian distributions) resulting from cutting the tree at a specified level. A *shortlist* is a set of tree leaves having a common *codeword*.

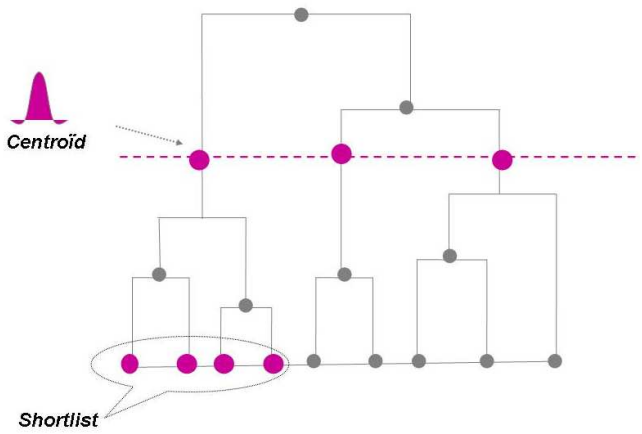


FIG. 2. Gaussian distributions classification

2.2 Model shortening

As a first application of the clustering algorithm, we propose to apply it to model shortening. Then we investigate its impact on the system performance.

2.2.1 Experimental setup

For all the experiments we use :

- parameter vectors with 12 MFCC coefficients, energy, and their first and second derivatives.
- 40 context independent models trained with 82 hours of the Ester train database (Galliano et al. (2005)).
- a dictionary containing 118000 words.
- a language model formed by 4 millions of bigrams and trigrams.
- an hour of Broadcast News extracted from the Ester test data set.

In order to compare the different systems, several reference systems 32, 64, 80, 128, 180, 256, 220 and 256 Gaussians per state are produced.

Considering the system with 256 Gaussians per state, the clustering algorithm is applied to each Gaussian mixture. Depending on the experiments, either the likelihood loss or the weighted cross entropy based metric is used. The previously tree cutting criteria (fixed, weight based or distance based) are also evaluated.

2.2.2 Fixed classes

The same number of classes is used by the reference system (REF) and by the loss likelihood (PV) and weighted Kullback-Leibler (KLP) based systems. After clustering, the PV and KLP models obtained are trained. We find that two iterations are sufficient for a good parameter estimation.Results within a confidence interval of 1% are as follows :

TAB. 1
WER for REF, PV, and KLP systems

Number of Gaussians	REF (%)	PV (%)	KLP (%)
32	42.6	40.6	39.5
64	40.4	38.0	37.5
80	38.3	37.4	36.9
128	37.3	36.2	36.2
180	36.4	36.1	36.2
220	36.3	35.8	35.5
256	36.3	-	-
512	35.5	-	-

Table 1 and Figure 3 show that :
– Both PV and KLP systems outperform the reference one.
– Performance of the KLP system using 220 Gaussians per state is similar to the reference with 512.
– WER decreases by about 3% compared to the reference system.
– With a large number of clusters differences are less noteworthy.

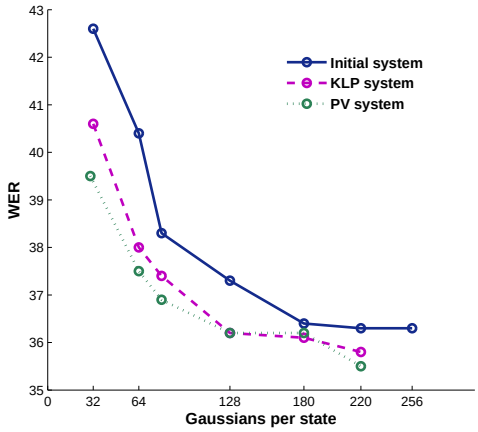


FIG. 3. WER vs number of Gaussians, for REF, PV and KLP systems

2.2.3 Weight based classes

By using this criterion, we ensure that each cluster has a sufficient amount of training data to estimate it. As the number of Gaussians per state is variable (it depends on the acoustic variability of each state), a mean value is considered. Results are as reported in Table 2 and Figure 4.

TAB. 2
PV and *KLP* weight based cutting

Metric	Number of Gaussians	*WER* (%)
KLP	28	40.0
	53	36.6
	150	35.9
	195	36.0
PV	53	39.5
	101	36.8
	156	36.5

We notice *KLP* outperforms both the *PV* and the reference system. Especially, with a mean of only 53 Gaussians per state, its performance is close to that of the reference system with 256 Gaussians per state. Besides, the *WER* decreases by about 4.8% compared to the initial system using the same number of Gaussians.
28 Gaussians per state on the *KLP* system perform better than 64 Gaussians in the reference system.

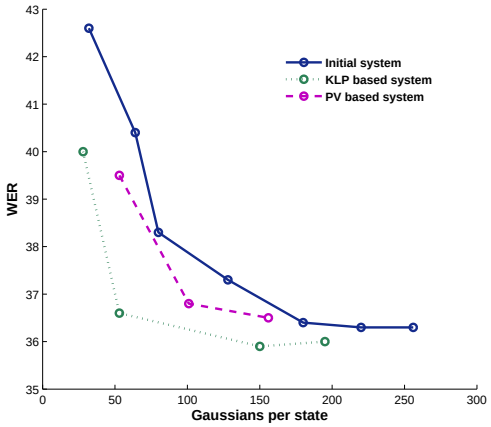


FIG. 4. Weight based tree cutting

2.2.4 Distance based classes

This criterion prevents the clustering of too distant Gaussians. Gaussians are considered distant when they are too different (in the case of KLP distance) or when merging leads to a big likelihood loss (if PV based metric is employed).

We consider several levels of the tree and cut when the distance between two levels reaches a maximum value. The obtained results are reported in table 3.

TAB. 3
PV and KLP distance based cutting

Metric	Number of Gaussians	WER (%)
KLP	30	40.7
	59	37.7
	101	36.1
	196	35.9
PV	44	39.4
	94	36.7
	204	35.8

Once again, we notice that PV and KLP systems outperform the reference, and that the KLP divergence based system is the best. Applying KLP or PV clustering process, we obtain globally the same performance as the reference system using only about 40% of the total number of Gaussians. These results are good but they remain not as good as the previous experiments (53 Gaussians) in which only 20% were used.

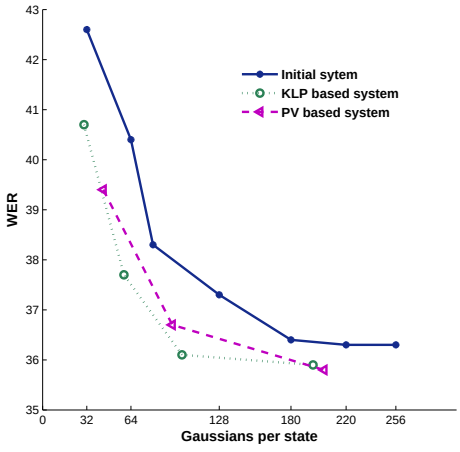


FIG. 5. Distance based tree cutting

2.2.5 Weight versus distance

To compare distance and weight criteria we plot their curves using both KLP and PV metric.

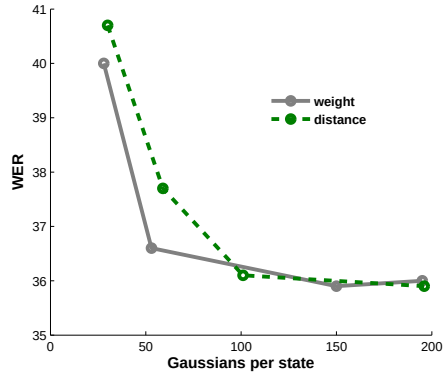


FIG. 6. Weight and distance based tree cutting for the KLP system

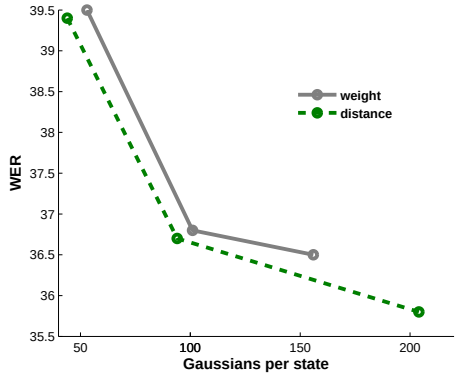


FIG. 7. Weight and distance based tree cutting for the PV system

For the KLP system, the weight criterion performs better than distance, especially when the number of clusters is low. In the case of PV clustering, the situation is the contrary. These results can be interpreted as follows :

– When the KLP clustering metric is used, no particular attention is given to the amount of training data available for each cluster. Only resembling Gaussians are merged, ensuring that at each level clusters are as distant as possible. So in some levels many clusters may not have enough training data, and cutting at these levels is of little value.

– In the case of PV based clustering, the loss of likelihood is minimum at each level. So the resulting clusters are as representative as possible of the training data. Knowing that no information about similarity of clusters to each others is taken into account, many resembling clusters can be present in the same level. In this case the distance based cutting criterion can remove the redundant information.

2.3 State-based Gaussian selection

The second application of the classification algorithm is Gaussian selection. We investigate a likelihood-based and multi-level Gaussian selection.
The overall algorithm operates in two steps : in the first state Gaussians are organized into a binary tree. Several cuts of the tree are performed. Each level of cut is characterised by its number of codewords. Tree leaves having a common codeword form a shortlist.
In the second step (ie. selection) codeword likelihood is computed and sorted. Only the most likely codewords are considered when descending down to the lower level of cut. When the leaves of the tree are reached, the corresponding Gaussian distributions are sorted by weight and the best of them contribute to the likelihood computation.

2.3.1 The selection algorithm

Selection is applied during the decoding process. Its goal is to detect, for each node of the decoding graph, Gaussians that dominate the likelihood computation. It operates as follows :

(1) For the current level of cut, codeword likelihoods are computed. Then they are sorted and the most likely are kept before moving down to the lower level of cut.

(2) When reaching the last level of cut, 2 sets of Gaussian distributions can be selected for the likelihood computation :
   a) leaves whose ancestors have all been kept.
   b) leaves selected in a) with large weight values.

The following example (Figure 8) illustrates an application of this algorithm to a mixture of 24 Gaussian distributions. In this case two levels of cut are considered : level 1 and level 2.

First, likelihoods of the codewords 31, 32 and 33 are computed and sorted. As the codeword 31 is the most likely it is selected.
Then we move to the next level of cut (level 2) and compute the likelihood of the corresponding nodes that are 25 and 26. If codeword 26 is more likely, the corresponding leaves which are the Gausians 4,5, 6 and 7 are selected.
Finally we can decide to compute the likelihood with all of them or to keep only those with the highest weight values.
Hence we computed a total of 9 likelihoods which is less time consuming than 24.

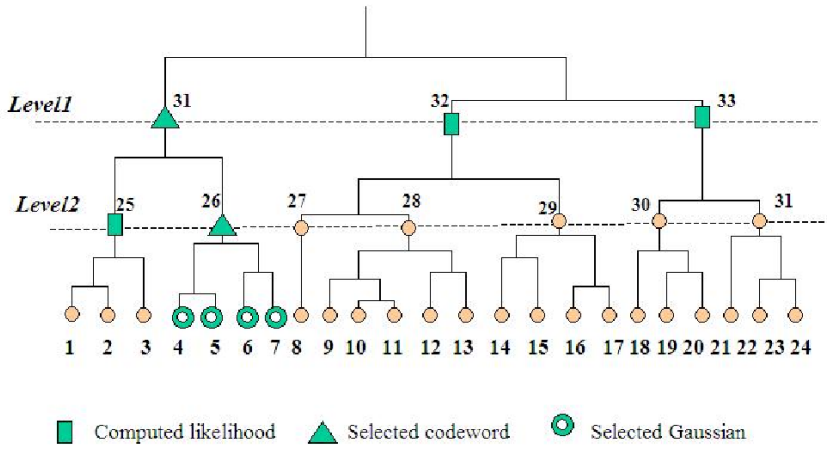


FIG. 8. Bi-level Gaussian selection example

2.3.2 Experimental setup

The Sirocco-Htk large vocabulary speech recognition system was used to compare the performance of the different schemes. This reference system makes use of 40 context independent acoustic models with 3 states each and 512 Gaussians per state. The parameters of these models are estimated on the Ester train database. The same parametrisation and textuel resources (dictionnary and language model) of the previous experiments are employed.
Tests are conducted using an hour of Broadcast News extracted from the Ester test data set.
The performance is addressed in terms of Word Error Rate (WER) and percentage of likelihood computation C. The latter is defined as :

$$C = \frac{computed \;\; likelihoods}{all \;\; likelihoods} \tag{6}$$

For the reference system : $WER = 35.5\%$ and $C = 100\%$

2.3.3 One-level based selection

For each state, the 512 Gaussian distributions are organized into a tree structure. We experimented cutting the tree at the levels 40 and 120 which correspond respectively to 40 and 120 codewords.

2.3.3.1 Shortlist scores : We vary the number of selected codewords and use the corresponding tree leaves for the likelihood computation.
For each number, the Gaussians and WER are reported in Figure 9. As the number of the selected Gaussians is variable, a mean value is considered. The fraction C is also computed and depicted in Figure 10.

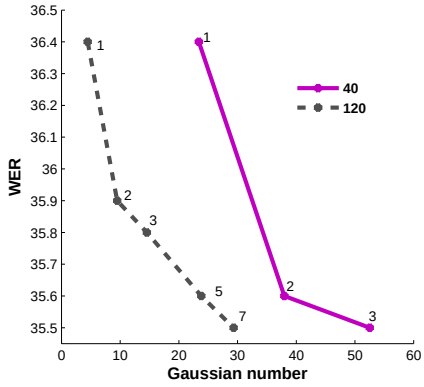


FIG. 9. Computing the likelihood using the best shortlists

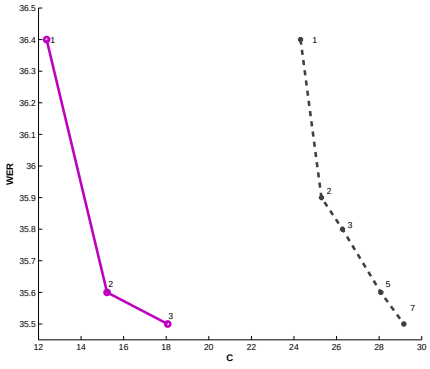


FIG. 10. Computing the likelihood using the best shortlists

For the same WER, the 120 codeword system makes use of fewerless Gaussians than the 40 codewords one. In particular, with only 23 Gaussians per state (Figure9) it gives exactly the same results as the reference system
The same experiments (Figure10) show that the value of C is lower for the 40 codeword system. This is because this fraction takes into account the codebook size. The best tradeoff between C and the WER is obtained by the selection of 2 codewords. This corresponds to the pair of values (C,WER)=(15.16%,35.6%). In this case the WER increases by only 0.1% and the likelihood computation cost is reduced by a factor of seven.

2.3.3.2 Data-based selection : We take the best system of the previous experiments : 40 codewords among which the 2 likeliest are selected.
As the training process is based on the "Maximum Likelihood" criterion, the likely distributions have large weight values. So, to reduce further the number of selected Gaussians, they are sorted by weight and only the components with highest weights are kept.
When varying the number of selected Gaussians, we obtain the results of Figure 11.

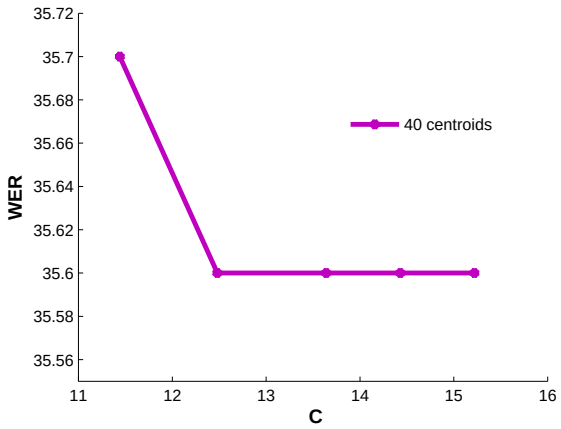


FIG. 11. Computing likelihood using the highest weight Gaussians

The best tradeoff between C and WER is (12.48%,35,6%). These results are better than those of the previous experiments. Indeed, for the same value of WER (35.6%) the value of C is reduced. In this case, the likelihood computational cost is decreased by a factor of eight.

2.3.4 Bi-level selection

Now the clustering tree is cut simultaniously into two levels of cut.Two bi-levels of cut are experimented : 40-60 codewords and 40-120 codewords.

2.3.4.1 Shortlist scores : In order to improve the results of the experiments in 2.3.3.1, all densities of level 40 are computed and the two best codewords are selected. Then we move to the second level of cut (that is 60 or 120). The corresponding codewords are computed and the most likely of them are kept. Finally, the Gaussians for there codewords are used for the likelihood computation.

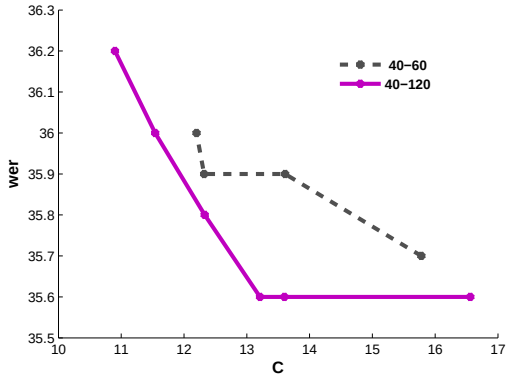


FIG. 12. Computing likelihood using two levels of cut and the best shortlists

The 40-120 system gives better results than 40-60. This is foreseeable because the level 120 is lower than the level 60 so the classification is more precise. The best tradeoff between C and WER corresponds to the pair of values (C,WER) = (13,21%,35.6%). This result is better than that in 2.3.3.1 where the tree was cut at a single level but less good than the result using weight values (2.3.3.2).

2.3.4.2 Data-based selection : We proceed in the same manner as in 2.3.4.1. The best settings are considered : bi-level of cut 40-120, and the best pair of values (C,WER)= (13,21%,35.6%). The optimization of the system consists in keeping only the Gaussians with the highest weight values. When varying the number of selected Gaussians, we obtain the results of Figure 13.

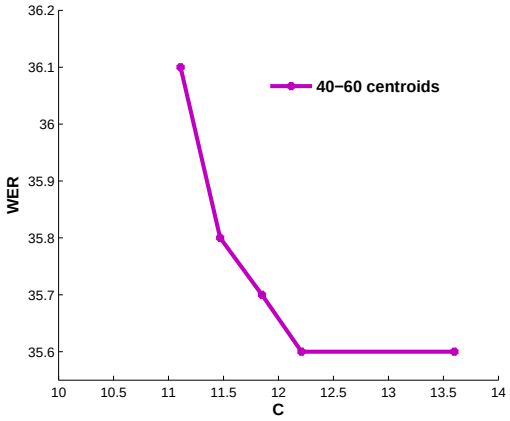


FIG. 13. Computing likelihood using the hightest-weight Gaussians

Here, (C,WER)= (35,6%, 12.20%) is the best tradeoff between C and WER. As the width of the confidance interval is about 0.8%, other tradeoffs are also satisfactory. It is for example the case of the pair of values (C,WER)= (35,8%, 11.5%) which corresponds to a decrease in the likelihood computation cost by a factor of nine with a non significant loss of accuracy (+0.3%).

2.3.5 Synthesis

Method	C (%)	$+WER$(%)
one level	15.16	0.1
two levels	13.24	0.1
one level + weight	12.48	0.1
two levels + weight	12.20	0.1

a- Bi-level selection is better than one-level selection. So increasing the number of levels is advantageous.

b- The selection of Gaussians with high weight values improves the performance.
c- The combination of (a) and (b) gives the best results.

3 Model based clustering and selection

To reduce the likelihood cost in HMM based ASRs two main approaches are generally used : Gaussian selection and Sub-vector quantization. In both cases, the classification (Bocchieri (1993); Ortmanns et al. (1998); Gales et al. (1999); Padmanablan et al. (1999)) is performed by clustering/merging Gaussian distributions. So the contextual information is lost and some distributions will be assigned to codewords of different context.
Hence we propose a context based classification method. The idea is to use Gaussian distributions of context independent models as codewords. Then the multi-level classification algorithm is applied as a further improvement of the codebook. Indeed, this process provides a compact and more efficient codebook.
Each obtained codebook is tested within the framworks of Gaussian selection and sub-vector quantization experiments.

3.1 Contextual Gaussian selection

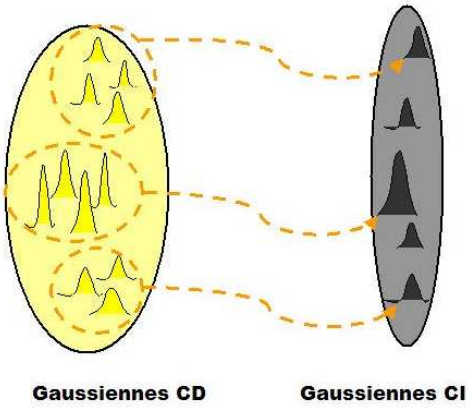
Initially, Bocchieri (Bocchieri (1993)) proposed a Gaussian selection technique by vector quantization. He generates a vector quantized codebook and attributes a shortlist to each codebook entry. During decoding, the frame is assigned to the nearest codeword/shortlist. Gaussians belonging to this shortlist contribute to that frame likelihood computation. Later, many extensions of this work have been proposed in the literature (Gales et al. (1999); Olsen (2000); Leppänen et al. (2006)). They were focused on Gaussian assignments to classes. Here we are rather interested in improving class building and more precisely their centroids.

3.1.1 CD-CI mapping

Large vocabulary continuous speech recognition systems need a significant number of Gaussians to model the different contexts. So Context dependent (CD) models are often used and the corresponding systems are generally slow. Small vocabulary systems make use of context independent models (CI) because the acoustic variability is limited. Hence, they have the advantage of being fast.

As both models CD and CI when trained on the same data should represent the same context, we investigate the use of CI Gaussian distributions as codebook in a CD model based large vocabulary ASR system.

The proposed contextual Gaussian selection method consists in :
– Computing all the distances between the CI and the CD Gaussian distributions.
– Assigning each CD Gaussian distribution to the nearest CI.
– Mapping CD-CI distributions.



Hierarchical clustering of Gaussian distributions

Hence the codebook contains the CI Gaussian distributions. The set of CD distributions assigned to the same codeword (CI Gaussian distribution) form a shortlist.

3.1.2 Hierarchical mapping

The state based clustering algorithm (subsection 2.1) is applied to the CI distributions in order to reduce their length and improve their representation. Let us keep in mind that the idea behind this procedure is to explore a large set of distributions and to reduce it by merging the closest components. A mapping table between the CD distributions and the new codebook (CI distributions after the clustering process) is constructed.

3.1.3 Experimental setup

We create a large vocabulary speech recognition system based on Sphinx training/test tools. The acoustic models are cross-word CD with 6108 tied states and 32 Gaussians per state. The parameters of these models are estimated on the Ester train database. Tests are conducted using an hour of Broadcast News extracted from Ester test set. For this reference system all the Gaussian

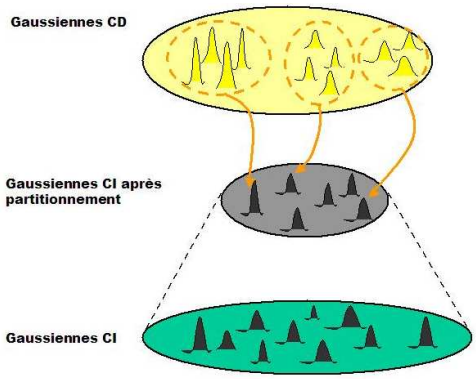


FIG. 14. Mapping between the new CI et CD distributions

distributions are used for the likelihood computation and so C=100%. The WER is equal to 28.7%.

3.1.4 CI-based codebook

36 context independent models with 32 Gaussians per state and 3 states each are developed. They have a total of 3456 Gaussian distributions (36*3*32). After mapping the CD distributions to CI, we apply the classical Gaussian selection procedure Bocchieri (1993). We also apply the classical Gaussian selection method to a codebook obtaind by clustering for comparative raisons. We report in the Figure15 WER and C corresponding to varying the number of selected Gaussians.

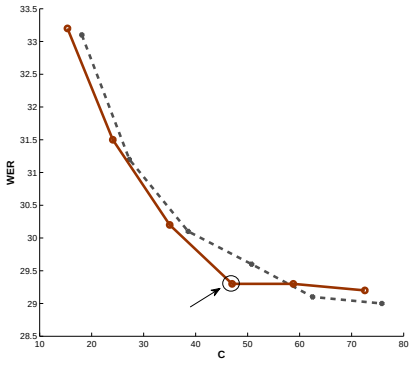


FIG. 15. WER vs C for classic and contextual Gaussian selection

We can see that contextual performs better than classic Gaussian selection. The best tradeoff between WER and C corresponds to the pair of values (29.3%,47.02%) and a loss of accuracy of 0.6% absolute.

3.1.5 Hierarchical codebook

Hierarchical clustering is applied to CI models with 64 and 128 Gaussians per state to be reduced to 32 Gaussians per state. Then classic Gaussian selection is performed using the new codebook.

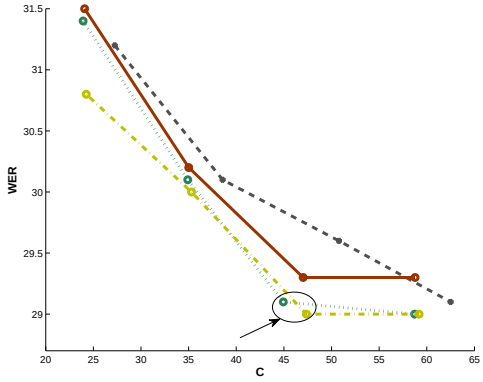


FIG. 16. WER vs C for classic, contextual and hierarchical Gaussian selection

From the Figure 16, we notice that the curves of the models initially with 64 and 128 Gaussians per state are the lowest. Typically, with 47% of computed likelihood the loss of accuracy is less than 0.3%.

3.2 Contextual sub-vector quantization

Recently, sub-vector quantization based methods were proposed as an alternative to the Gaussian selection approach. They have been successfully applied to reduce the acoustic model complexity without a significant loss of accuracy. Several methods of codebook construction have been investigated. They are generally based on clustering techniques. For example Mak (Mak et al. (2001)) performs per stream Gaussian clustering by means of Battacharya distance. A speech group at Carnegie Mellon University employs k-means algorithm to cluster sub-vectors (means and variance) into a preset number of codebooks (Ravishankar et al. (1997)), ..
As the contextual information is lost by clustering, we are interested (in this subsection) in contextual sub-vector quantization. Then, we investigate the improvement of the codebook by hierarchical clustering.

3.2.1 Stream-based mapping

The contextual sub-vector quantization method is performed in three steps :

(1) the mean and variance vectors of each context independant (CI) distribution are divided into streams i.e. subsets of dimensions.

(2) the context dependent distributions (mean and variance vectors) are divided into the same dimension subsets.

(3) for each stream, the symmetric Kullback-Leibler distances between the CD and CI distributions are computed. Each CD distribution is assigned to the closest CI distribution.

By the end of this process, we obtain a per stream mapping table between the CD and CI distributions.

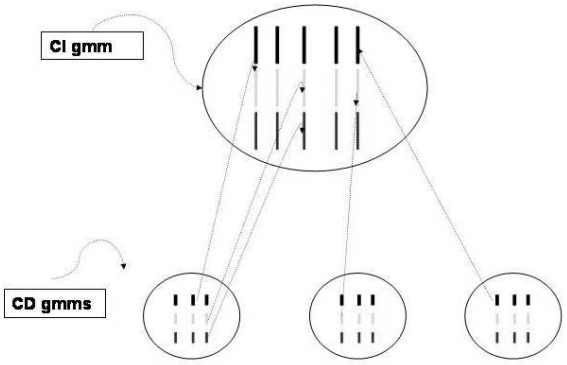


FIG. 17. Per stream mapping between CI and CD distributions

3.2.2 Experimental setup

In all of the experiments, the performance of the proposed methods are compared to those of the initial system which are $(WER, C) = (28.7\%, 100\%)$ and to those of an existing sub-vector quantization method (described in Ravishankar et al. (1997) and 3.2). Let us keep in mind that the existing sub-vector quantization (SVQ) method is based on a simultaneous mean and variance vectors clustering to construct the codebook. The K-means algorithm is employed to perform the clustering process by stream.

The parameter vectors are composed of (12mfcc + energy) and their first and second derivatives. Three subdivisions of the parameters vector are tested :
– Only one stream of dimension 39 is considered.
– Three streams : (12mfcc+energy) + Δ(12mfcc+energy)+ Δ Δ(12mfcc+energy).
– Four streams : (energy,Δenergy,Δ Δenergy) + (12 mfcc) + (12 Δ(mfcc)) + (12 Δ Δ (mfcc)).

3.2.3 Contextual Sub-Vector Quantization

For the contextual sub-vector quantization (CSVQ), 36 CI models with 32 Gaussians per state (a total of 108 states) are used for the codebook. The mapping is performed by state (CSVQ-s), by phone (CSVQ-p) or using all the CI distributions (CSVQ-a).

- CSVQ-s : the Gaussian distributions of each CI state constitute a codebook for the corresponding triphone states.
- CSVQ-p : the Gaussian distributions of each CI phone constitute a codebook for the corresponding triphones.
- CSVQ-a : all the CI distributions constitute a codebook for all the triphone states.

During the decoding process, the likelihood is computed with the CI distributions and the corresponding CD distribution weights.

Figure 18 reports the WER according to the computation fraction C for the methods SVQ, $CSVQ-s$, $CSVQ-p$ and $CSVQ-a$ and for the three sizes of stream.

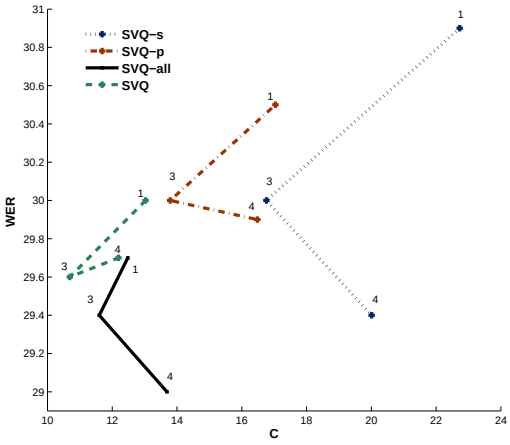


FIG. 18. The performance of the SVQ and CSVQ methods.

Several remarks can be made :

- Multi-stream based methods give better results than those with only one stream which is logical (there is less quantification distortion).

- The increase in WER generated by the SVQ, $CSVQ-s$ and $CSVQ-p$ methods exceeds 1%. From our point of view, this is due to the loss of information about context in the SVQ method. For the $CSVQ-s$ and $CSVQ-p$ approaches we can say that the distributions of the CI states

and phones are unable to represent all the corresponding CD distributions.

– The best results are obtained by means of the CSVQ-a method. We point out that this method makes use of a codebook formed by all of the CI distributions. The optimal configuration corresponds to the pair of values $(WER, C) = (29.0\%, 13.69\%)$, i.e. the likelihood computation fraction is reduced to 13.69% with a small increase of the WER (+0.3% absolute).

3.2.4 Multi-level Contextual Sub-Vector Quantization

To improve the results of the $CSVQ - a$ method, we applied the hierarchical clustering algorithm to the CI distributions. The initial codebook is formed by 3456 (ie. 32*108) CI distributions. One level of tree is considered : 5 Gaussians per state. This level corresponds to the final codebook containing 540 distributions. To compare the results of these experiments to the previous ones, we use the same stream definitions. We also apply the SVQ method with the same codebook length.

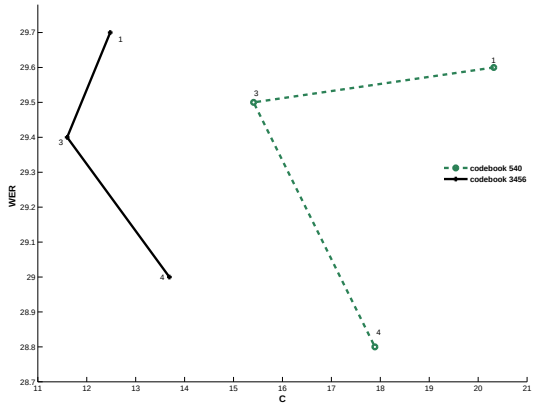


FIG. 19. The SVQ and CSVQ-multi-level results for one stream

From Figure 19, we can notice that :

– The CSVQ-multi-level method outperforms SVQ.

– Using only one stream is of little interest (the WER increase exceeds 0.9% absolute).

– The CSVQ-multi-level method produces an interesting point (WER,C)=(28.8%,17.89%) which compares well with the initial system. In addition, the WER is inside the confidence interval and thus we can conclude that about 17% of densities are computed with no loss of accuracy.

4 Conclusion

This paper presents several algorithms to reduce the computation cost in low-resource and large application mobile devices. The proposed Gaussian selection and sub-vector quantization techniques aim to decrease the cost of likelihood computation both in the state level model in the model one.
Evaluation has been performed within the framework of a large vocabulary continuous speech recognition task. The likelihood computation cost is reduced to 17% with no loss of accuracy. As a perspective, we propose to combine this technique with a Gaussian selection method.

5 Acknowledgments

The authors would like to thank Peter Weyer-brown for his help.

Références

Bocchieri, E., Vector Quantization for the Efficient Computation of Continuous Density Likelihoods, 1993, International Conference on Acoustics Speech and Signal Processing, 692-695.

Digalakis, V., Tsakalidis, S., Harizakis, c., & Neumeyer, L., Efficient Speech Recognition using Subvector Quantization and Discrete-Mixture HMMs, 2000, Computer Speech and Language, 33-46

Gales, MJF., McKnill, K., & Young, S., State based Gaussian Selection in Large Vocabulary Continuous Speech Recognition using HMMs, 1999, IEEE Transactions on Speech and Audio Processing, 470-473

Gales, MJF., McKnill, K., & Young, S., Use of Gaussian Selection in Large Vocabulary Continuous Speech Recognition using HMMs, 1996,International Conference on Spoken Language Processing, 470-473

Galliano, S., Geoffrois, E., Mostefa, D., Choukri, K., Bonastre, JF., & Gravier, G., The Ester Phase II Campaign for the Rich Transcription of French Broadcast News, 2005, European Conference on Speech Communication and Technology

Jurgen, F., & Ivica, R., The Bucket Box Intersection (BBI) Algorithm for Fast Approximative Evaluation of Diagonal Mixture Gaussians, 1996,International Conference on Acoustics Speech and Signal Processing, 837-840

Jurgen, F., Ivica, R., & Tile, S., Speeding up the Score Computation of HMM Speech Recognizers with the Bucket Voronoi Intersection Algorithm, 1996, European Conference on Speech Communication and Technology,1091-1094

Leppanen, J., & Kiss, I., Gaussian Selection with Non-Overlapping Clusters

for ASR in Embedded Devices, 2006, International Conference on Acoustics Speech and Signal Processing

Mak, B., & Bocchieri, E. Subspace Distribution Clustering Hidden Markov Model, 2001, IEEE transactions on Speech and Audio Processing, 264-275

Mokbel, C., Online Adaptation of HMMs to Real Life Conditions : A Unified Framework, 2001, IEEE Transaction on Speech and Audio Processing, 342-357

Olsen, J., Gaussian Selection using Multiple Quantisation Indexes, 2000, IEEE Nordic Processing symposium

Ortmanns, S., Ney, H., & Firslaff, T., Fast Likelihood Computation Methods for Continuous Mixture Densities in Large Vocabulary Speech Recognition, 1998, European Conference on Speech Communication and Technology, 139-142

Padmanablan, M., Bahl, L., & Nahamoo, D., Partitioning the Feature Space of a Classifier with Linear Hyperplanes, 1999, IEEE Transactions on Speech and Audio Processing, 282-288

Ravishankar, M., Bisiani, R., & Thayer, E., Sub-vector Clustering to Improve Memory and Speed Performance of Acoustic Likelihood Computation, 1997, European Conference on Speech Communication and Technology

Woszczyna, M., Fast Speaker Independent Large Vocabulary Speech Recognition, 1998, PHD Thesis, Karlsruhe University