Acoustic Phonetic Modelling using Local Codebook Features

Frank Diehl, Asunción Moreno

TALP Research Center, Universitat Politècnica de Catalunya (UPC)
Jordi Girona 1-3, 08034 Barcelona, Spain
{frank,asuncion}@gps.tsc.upc.es

Abstract

In this article we present an alternative method for defining the question set used for the induction of acoustic phonetic decision trees. The method is data driven and employs local similarities between the probability density functions of hidden Markov models. The method is shown to work at least as well as the standard method using question sets devised by human experts.

1. Introduction

A central question in the design of an automatic speech recognition (ASR) system is the definition of proper acoustic phonetic entities. For a state of the art ASR system context dependent models as e.g. triphones are used. A drawback of these models is their theoretically high number, making it impossible to train all of them sufficiently or, not at all. A common solution consists of clustering the seen models in groups by the use of a phonetic decision tree. Basically, this serves for two purposes. From a training point of view, it defines the final hidden Markov models (HMM) to be trained. That is, it serves as a classification tree using unsupervised clustering to group the seen HMMs to trainable entities. On the other hand, from a recognition point of view, we use it as a regression tree, finding a proper HMM for each possible acoustic phonetic circumstances. That is, we exploit the functional mapping from the feature to the model domain the tree represents.

One crucial topic of this mapping function is the definition of the input domain. A standard approach is the use of IPA-based acoustic phonetic features assigned to the phonemes. Even if this usually works quite good, it exhibits the problem that the designer is dependent on phonemical knowledge. In case of a foreign or unknown language this might be a severe problem. To cope with this, several methods were proposed to construct the features data driven. Ciprian et al. [1] present an approach based on a bottom-up clustering using bigram phone statistics. Beulen et al. [2] also propose a bottom-up approach but already incorporate acoustic information by making use of the HMMs seen in the database. Both methods use all or a subset of the nodes in the agglomerative constructed trees as features of the input domain. As a third approach, although more related to multilingual phonetic modelling, we mention the construction of phonetic broad classes by a confusion matrix, Byrne et al. [3].

In this work, we present an alternative solution to the problem. The basic idea is to search for local similarities between the probability density functions of HMMs. For doing so, we exploit the prototype character of the single mixture components of the codebooks of semicontinues HMMs (SCHMM). As will be shown in the following, the formulation of the problem by SCHMMs allows us to reduce the search problem to simple vector calculus. Because the features are based on local similarities within the codebooks they are named ”local codebook (LCB) features”.

This work was granted by the CICYT under contract TIC2002-04447-C02.

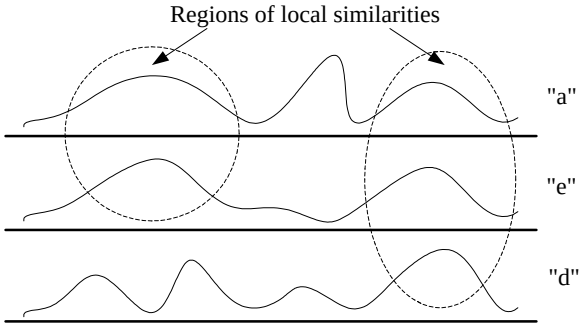

Figure 1: *Local codebook similarities.*

The paper is organized as follows. In Section 2 the LCB features are derived, followed by Section 3 introducing the decision tree. Question generation is discussed in Section 4. Section 5 gives a system overview followed by Section 6 with the test set up. Test results are presented in Section 7 and the conclusions are given in Section 8.

2. Local codebook features

The basic idea behind LCB-features is that similar phonetic properties should cause similar shaped probability density functions (PDF) on a local scale. In Figure 1, this idea is depicted. It shows the assumed probability density functions of the phonemes ’a’, ’e’, and ’d’, with two locally similar regions. The similarities might be caused by common properties of the phonemes as e.g. ’voiced’ or ’open’.

Constructing features based on this idea means to identify such similarities in the PDFs of HMMs. In the case of continues HMMs, this might be a quite hard task, but in the case of SCHMMs, it results in simple vector calculus. Assuming the PDFs being constructed by a set of prototype mixture components, the calculation can be reduced to the prototype weights. For the sake of simplicity, we assume during derivation of the method, that the beforehand trained incontextual HMMs have only one state and refer to only one codebook. In the following, this allows to leave out the state and codebook indices. Hence, the PDF of one SCHMM is given as

$$G_{mix}(i) = \sum_{l=1}^{L} c_{il} \cdot G\left(\mu_l, \cdot\right) \qquad i \in \{1, ..., I\}, \qquad (1)$$

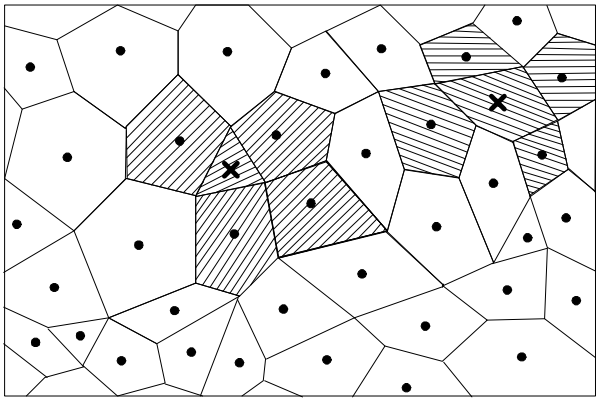

Figure 2: *Locality and neighborhood.*

with L the codebook size and i the HMM index. $G(\mu_l, \cdot)$ names the l^{th} mixture component with mean vector μ_l. For the local search, we define a locality just by a mixture component, and therefore by every index $\tilde{l}$ out of the L mixture components. I.e. there are L localities. We also define the neighborhood of $\tilde{l}$ as the $\tilde{L}$ mixture components closest to the locality $\tilde{l}$.
Figure 2 depicts the concept for a two dimensional case. It shows a part of a codebook with class regions and the mean values (black dots). Two localities are accentuated by marking the corresponding means by crosses instead of dots. The neighborhood is set to $\tilde{L} = 5$, and the equivalent regions with the five closest mean values to the localities are drawn hatched.
For a formal derivation of the method, we define the distance $d(\tilde{l}, l)$ between the mixture components by Equation (2)

$$d(\tilde{l}, l) = \langle \mu_{\tilde{l}}, \mu_l \rangle \qquad l, \tilde{l} \in \{1, ..., L\}. \tag{2}$$

Identifying for each locality $\tilde{l}$ the $\tilde{L}$ closest neighbors is done by evaluating Equation 2 for all l and $\tilde{l}$. As result we get for each locality $\tilde{l}$ an index set $S_{\tilde{l}}$ naming the indices of the $\tilde{L}$ closest mixture components of locality $\tilde{l}$. Using $min^{(n)}$ to signify the "n^{th} smallest value of" we can express $S_{\tilde{l}}$ as

$$S_{\tilde{l}} = \left\{ l \mid \arg \min_{1 \le l \le L}{}^{(n)} d(\tilde{l}, l), \quad n \in \left\{1, ..., \tilde{L}\right\} \right\}, \tag{3}$$

where $\tilde{l} \in \{1, ..., L\}$.

Applying these index sets to the PDF for each HMM i, it results into local weights vectors $\tilde{c}_{\tilde{l}i}$ comprising of each HMM i the weights of the codebook mixture components in the neighborhood of the locality $\tilde{l}$. Hence, $\tilde{c}_{\tilde{l}i}$ represents the local shape of the underlying PDF. Although this is not completely correct, the remaining mixture components also contribute to the probability mass at the locality $\tilde{l}$, it is a reasonable simplification. Grouping all local weights vectors of a locality $\tilde{l}$ together, we get the $\tilde{L}$xI matrix $\tilde{C}_{\tilde{l}}$

$$\tilde{C}_{\tilde{l}} = [\tilde{c}_{\tilde{l}1}, ..., \tilde{c}_{\tilde{l}I}]\,. \tag{4}$$

In this vector formulation, the term "locally similar" for the PDFs of different HMMs is equivalent to vectors $\tilde{c}_{\tilde{l}i}$ being close together in the vector space spanned by them. Searching for local similarities is therefore reduced to the search of suitable clusters in the space $span\{\tilde{c}_{\tilde{l}1}, ..., \tilde{c}_{\tilde{l}I}\}$. This search can be performed in a number of ways. For our experiments, we used a quite simple method based on a singular value decomposition (SVD). The basic idea is to identify the principle components of the samples defined by $\tilde{C}_{\tilde{l}}$ and to group together the $\tilde{c}_{\tilde{l}i}$ participating mostly to the strongest principle components. Hence, we calculate

$$\tilde{C}_{\tilde{l}} = U_{\tilde{l}} \Sigma_{\tilde{l}} V_{\tilde{l}}^T \tag{5}$$

with

$$U_{\tilde{l}} = [u_{\tilde{l}1}, ..., u_{\tilde{l}\tilde{L}}] \tag{6}$$

being the principle components, and

$$\Sigma_{\tilde{l}} = [\sigma_{\tilde{l}1}, ..., \sigma_{\tilde{l}\tilde{L}}] \tag{7}$$

the singular values. We record that the sum of the squared projection lengths of all $\tilde{c}_{\tilde{l}i}$ on a principal component is just given by the corresponding squared singular value. E.g. for the strongest principal component $u_{\tilde{l}1}$ we have

$$\left\| u_{\tilde{l}1}^T \tilde{C}_{\tilde{l}} \right\|^2 = u_{\tilde{l}1}{}^T \tilde{C}_{\tilde{l}} \tilde{C}_{\tilde{l}}^T u_{\tilde{l}1} = \sigma_{\tilde{l}1}^2. \tag{8}$$

With the components of $\tilde{c}_{\tilde{l}i}$ being the probability masses of the affected PDF mixture component of HMM i, a singular value is directly related to the probability mass associated to the corresponding principal component. This relationship is exploited to define suitable clusters being equivalent to the desired features. We introduce threshold T_{Class} and T_{Member} with:

T_{Class} The maximum amount of squared singular value mass to cover by the used principal components.

T_{Member} The maximum amount of squared singular value mass per used principal component.

$T_{Class} \in [0, 1]$ controls the number $\tilde{L}'$ of classes we might extract per locality $\tilde{l}$. We take as many of the strongest principal components into account until

$$\frac{\sum_{l=1}^{\tilde{L}'} \sigma_{\tilde{l}l}^2}{\sum_{l=1}^{\tilde{L}} \sigma_{\tilde{l}l}^2} \tag{9}$$

exceeds T_{Class}. Singular value mass related to the remaining principal components is assumed to be less informative.
$T_{Member} \in [0, 1]$ controls the number I' of local weights vectors $\tilde{c}_{\tilde{l}i}$ falling in a class, and, by the HMM-index i, which phonemes contribute to the cluster. That is, we construct a specific cluster $\tilde{l}' \in \{1, ..., \tilde{L}'\}$ by assigning more and more of the strongest $\tilde{c}_{\tilde{l}i}$ to the cluster until

$$\frac{\sum_{i=1}^{I'} \left\| u_{\tilde{l}l'}^T \tilde{c}_{\tilde{l}i} \right\|^2}{\sigma_{\tilde{l}l'}^2} \tag{10}$$

exceeds T_{Member}. The classes with the corresponding HMMs constitute the desired local codebook features.
In practice, the feature extraction described here is done on incontextual HMMs having more than one state with several sub-feature. The multitude of states simply lead to more local weights vectors $\tilde{c}_{\tilde{l}i}$ improving the estimation of the codebook features. Also several sub-features do not pose a problem to the method. We might just use one of them or construct a set of features for each of them , merging these sets afterwards to one global feature set altogether.

3. The decision tree

We use a binary decision tree with an entropy related impurity measure. It starts by pooling together all models of the training set to a median model for the root node t. Since a SCHMM is defined by its class weights $c_{i,j,k,l}$, where i stands for the HMM, j for the state, k for the sub-feature, and l for the codebook class, new, median class weights $c_{t,j,k,l}$ are calculated. They are determined as weighted mean over the corresponding class weights of all models i assigned to node t

$$c_{t,j,k,l} = \frac{\sum_{\forall i \in t} N_i \cdot m_{i,j} \cdot c_{i,j,k,l}}{\sum_{\forall i \in t} N_i \cdot m_{i,j}}. \qquad (11)$$

N_i is the number of times model i is seen in the training, and $m_{i,j}$ gives the expected number of frames the training stays in state j when being in model i. The product $N_i m_{i,j}$ defines the expected number of frames the training stays in state j of model i.

The algorithm goes on by searching the best split for the root node. For all possible binary questions Q assigned to the root node, entropy based impurity measures $i(Q, t_L)$ and $i(Q, t_R)$ for the left and the right child node as well as for the root node itself are calculated. The impurity $i(t)$ of node t is defined as

$$i(t) = -N_t \sum_{\forall j} \frac{m_{t,j}}{m_t} \sum_{\forall k} \sum_{\forall l} c_{t,j,k,l} \cdot \log_2 c_{t,j,k,l}. \qquad (12)$$

The value m_t reflects the expected number of frames the training spends in a model related to node t and $N_t \frac{m_{t,j}}{m_t}$ gives the expected number of times the training sees node j when staying in a model of node t. $i(t)$ changes to $i(\cdot, t_L)$ and $i(\cdot, t_R)$ by reducing the preceding calculation of $c_{t,j,k,l}$ to the models assigned to the left and right child node of t, respectively.

Finally, the question Q^* giving the maximal decrease in impurity when splitting the root node is determined by

$$Q^* = \arg\max_Q \{i(t) - i(Q, t_L) - i(Q, t_R)\}. \qquad (13)$$

The split defined by Q^* divides the HMMs into two subsets. As the result, we get two new nodes, the children of the root.

The process of splitting a node is repeated for each new child node until a stopping criterion is met, or a node can not be split anymore. This may be due to the fact that no more questions are left or the questions left can not split the remaining HMMs into two non empty subsets. As stopping criterion a minimal training count is used.

Even though the formulation of the algorithm applies to model tying, it is also capable of state and sub-feature tying. In order to perform the necessary calculations on state and sub-feature level, proper modifications of formula 11, 12, and 13 are applied.

4. Question generation

As mentioned in section 3, a node is split according to the best binary question Q^* found for this node. In the classical approach, a binary question is composed out of phonetic attribute values assigned to the SAMPA representations of the phonemes we use in our system. The attribute values are taken from corresponding IPA descriptions [6] of the phonemes. Examples are given in Table 1.

A binary question is composed out of one up to two phonetic attribute values associated with the context of a model. According to Table 1, a question may be: "Is the context of the model plosive or close?". This is done for all possible compound questions of an attribute and for all attributes.

Table 1: *Examples of phonetic attributes.*

	p	b	i	e
Attri. 1	consonant	consonant	vowel	vowel
Attri. 2	obstruent	obstruent	front	front
Attri. 3	plosive	plosive	close	close-mid
Attri. 4	-	-	short	short

In case of LCB features, we started by training incontextual models for each language. This was followed by extracting the features as described in Section 2. The features are based on the MFCC-codebooks with $T_{Class} = 0.5$ and $T_{Member} = 0.4$. The neighborhood was set to $\tilde{L} = 6$. We found 211, 243 and 206 classes for German, UK-English and Spanish respectively. A selection of the feature classes extracted for German is shown in Table 2. These feature classes have to be transformed into a suitable form for the decision tree. Table 3 shows this exemplarily for the LCB features number 1, 2 and 3 of Table 2. A question corresponding to Table 3 is: "Does the context of the model belong to attribute class 4?".

Table 2: *German LCB feature classes.*

LCB feature	Class member
1	S Z
2	S Z dZ tS
3	m l
4	2: 9 @ E l OY
5	9 E 6 @ l s o~
6	Y f pf s ts

Table 3: *German LCB features.*

	S	Z	dZ	tS	m	l
Attri. 1	X	X	-	-	-	-
Attri. 2	X	X	X	X	-	-
Attri. 3	-	-	-	-	X	X

5. System overview

The system we use works with SCHMMs. Every 10ms twelve mel-cepstrum coefficients (MFCC) (and the energy) using cepstral mean subtraction are computed. First and second order differential MFCCs plus the differential energy are employed. For each sub-feature, a codebook is constructed consisting of 256 and 32 (delta energy) gaussian mixtures, respectively.

Acoustic phonetic modelling is done by using demiphones, Mariño et al. [4]. Demiphones can be thought of as triphones which are cut in the middle giving a left and a right demiphone. In contrast to triphones, they neglect the influence the left context of a phone might have on the right and vice versa. This drawback in its modelling capabilities is, at least partly, compensated by its improved trainability due to the reduced number of models. Assuming N phonemes, we get N^3 possible triphones, but only $2N^2$ demiphones.

The training process we apply, basically consist of 4 steps.

- Training a set of initial incontextual demiphones. This set is used for the initialization of all contextual demiphones and the extraction of the codebook features.

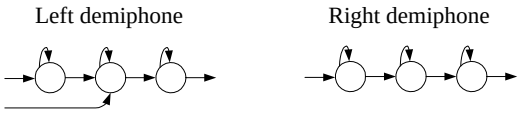

Figure 3: *Demiphone topology.*

- Training an intermediate set of contextual demiphones used for training the decision tree.
- Training the decision tree.
- Training the final contextual demiphones as defined by the leaves of the decision tree.

The model topology we use is depicted in Figure 3. Only for the three garbage models and the silence model four states are used.

6. Test set up

Training and testing the systems were performed using the 4000 speaker SpeechDat-II fixed telephone databases. From each database, a 1000 speaker training and a 400 speaker test part were extracted. For training, we used phonetical rich sentences. For testing, phonetical rich words mixed with application words were used. The test suit consists of isolated words avoiding the need of a language model. Detailed statistics on the corpora are given in Table 4.

	Phrases	Females	Males	Grammar size
ES-Training	7994	500	500	-
ES-Test	2644	200	200	1438
EN-Training	8089	500	500	-
EN-Test	2554	200	200	1254
GE-Training	7540	500	500	-
GE-Test	2700	200	200	1314

Table 4: Training and test data statistics.

For each language, we trained a set of decision trees based on LCB and another set based on IPA features. That is, we built a decision tree for each Markov state of each base phone. With 47 phonemes for German, 44 for English and 31 for Spanish, we got 1128, 1056 and 744 trees for German, English and Spanish, respectively.
Each set of decision trees defined an initial set of the final HMMs. Afterwards, we trained additional model sets based on pruned trees to search for right sized trees.

7. Test results

Table 5, 6, and 7 present the word error rates (WER) resulting from the test set up described in Section 6. Each row compares

Table 5: *German test results.*

IPA-Test	WER [%]	LCB-Test	WER [%]
$State_{4789}$	9.67	$State_{4733}$	9.41
$State_{4200}$	9.96	$State_{4200}$	9.52
$State_{3600}$	9.89	$State_{3600}$	9.70

the WER of a two equally sized model sets based on IPA- and LCB-features. The subscripts within the tables name the number of states that the model sets comprise.

Table 6: *UK-English test results.*

IPA-Test	WER [%]	LCB-Test	WER [%]
$State_{3636}$	28.43	$State_{3590}$	26.86
$State_{3000}$	27.53	$State_{3000}$	26.94
$State_{2400}$	26.31	$State_{2400}$	27.29

Table 7: *Spanish test results.*

IPA-Test	WER [%]	LCB-Test	WER [%]
$State_{2079}$	6.58	$State_{2078}$	6.09
$State_{1950}$	6.32	$State_{1950}$	6.09
$State_{1800}$	6.13	$State_{1800}$	6.20

Comparing the results, we see that LCB-features mostly outperform IPA-features. Only in the case of English the overall best result was found for an IPA based model set.
A general problem for the evaluation is the search for the best operating point, i.e. of the right sized trees. It might have happened that one method outperforms the other just because we did not find the best pruned tree. Additionally, the quality of the used IPA features is hard to assess. We derived them manually from text books as best as possible, but their might be better ones. On the other hand, we need to bear in mind, that the SVD based method used to define the LCB feature classes is a quite simple one. Using more sophisticated methods for searching meaningful clusters might improve the quality of the LCB feature too.

8. Conclusions

In this work, we presented an alternative method to define phonetic features classes for decision tree based acoustic phonetic modelling. We demonstrated its performance for three languages. The method is fully data driven, exploiting local similarities of the PDFs of SCHMMs. This is in strong contrast to other methods, [1], [2], [3], which have a more integral character working on model or state level.
It needs to be emphasized that the resulting features have a general nature. Even though derived by exploiting the specific properties of SCHMMs, they may be used within the context of any other system architecture, e.g. for continues HMMs.
Future work will concentrate on more sophisticated methods to extract the attribute classes and on their use in a multilingual environment.

9. References

[1] Ciprian, C. and Morton R., “Mutual information phone clustering for decision tree induction”, ICSLP, 2002.

[2] Beulen, K. and Ney H., “Automatic question generation for decision tree based state tying”, ICASSP, 1998.

[3] Byrne, W. et al, “Towards language independent acoustic modeling”, ICASSP, 1998.

[4] J. B. Mariño et al, “The demiphone: an efficient contextual subword unit for continuous speech recognition”, Speech Communication, 2000

[5] M. Y. Hwang et al, “Predicting unseen triphones with senones”, IEEE Transactions on speech and audio processing”, 1996

[6] IPA, “Handbook of the International Phonetic Association”