DYNAMIC SHARINGS OF GAUSSIAN DENSITIES USING PHONETIC FEATURES

Kyung-Tak Lee, Christian J. Wellekens

Institut Eurécom
2229, route des Crêtes, B.P. 193
06904 Sophia Antipolis Cedex, France
{lee, wellekens}@eurecom.fr

ABSTRACT

This paper describes a way to adapt the recognizer to pronunciation variability by dynamically sharing Gaussian densities across phonetic models. The method is divided in three steps. First, a HMM recognizer outputs a lattice of the most likely word hypotheses given an input utterance. Then, the canonical pronunciation of each hypothesis is checked by comparing its theoretical phonetic features to those automatically extracted from speech. If the comparisons show that a phoneme of a hypothesis has likely been pronounced differently, its model is transformed by sharing its Gaussian densities with the ones of its possible alternate phone realization(s). Finally, the transformed models are used in a second-pass recognition. Sharings are dynamic because they are automatically adapted to each input speech. Experiments showed a 5.4% relative reduction in Word Error Rate compared to the baseline and a 2.7% compared to a static method.

1. INTRODUCTION

A given word is almost never pronounced in the same way, whether it is uttered by two speakers or by the same speaker at different times. Factors of this variability are numerous, for example gender, age and emotional state. While these differences of pronunciation do not affect much the speech understanding of a human listener, they on the contrary deteriorate performance of speech recognizers, especially in the case of spontaneous speech.

Explicit modeling of these phenomena proved its usefulness and made the recognizers more robust to pronunciation variability. The major contributions in this field were done at the lexical level, by adding new phone transcriptions to the basic lexicon and using various pronunciation models. An overview of existing methods is given in [1].

Modeling pronunciation variation is also possible at other parts of ASR systems, for example at the level of acoustic models. In this paper, we investigate the possibility of modeling it dynamically at the level of HMM states, with the hope of further increasing the recognition rate compared to a static method. The paper is structured as follows. Section 2 describes the concept used to model pronunciation variability at the level of acoustic models. Section 3 explains how this concept was used to model pronunciation variation in a static manner. Section 4 describes the extension brought to this method to make the modeling dynamic. Section 5 illustrates the experiments carried out to put these methods into practice, and is finally followed by a conclusion in section 6.

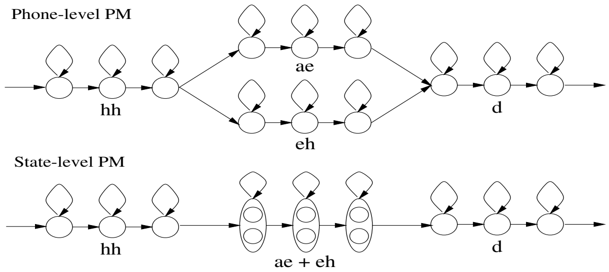

Fig. 1. Phone- vs. State-Level Pron. Modeling (from [2])

2. STATE-LEVEL PRONUNCIATION MODELING

We present in this section the concept used to model pronunciation variability at the level of acoustic models. Our work is mainly based on State-Level Pronunciation Modeling (SLPM) proposed by Saraçlar et al. [2]. The key idea is to model pronunciation variation by allowing sharings of Gaussian densities between phonetic models. Namely, if a phoneme may be realized into distinct phones, it can *share* their Gaussian mixtures with them so that it can inherit some of their acoustical properties as well. This is in contrast with the classical phone-level pronunciation modeling which supposes that only one of the distinct phonetic models may be used for a given case. The concept is illustrated in Figure 1 for the context independent case, where the word “had” may be pronounced canonically as “hh ae d”, but also differently as “hh eh d”. Consequently, the phonemes “ae” and “eh” may share their Gaussian densities state-by-state. When using context-dependent phones, the neighbouring phones follow the same rule, namely “hh” and “d” in their own right and left contexts.

3. STATIC SLPM

In order to see if a dynamic framework can bring better results than a static one, similar steps to those reported in [2] were followed to first build a static SLPM-based pronunciation model. It is assumed here that Gaussian mixtures are used as emission densities, and each density in the original system is supposed to belong only to a single state. The following steps are applied in the training phase :

1. Align the phonemic transcription of a sentence built from the lexicon with the hand-labeled phonetic transcription of the same sentence. We used a phoneme-to-phone alignment based on phonetic feature distances, then directly deduced the state-to-state correspondences.

2. From the alignment above, estimate the probability of a state b in the canonical transcription to be aligned to a state s in the surfacial transcription : $P(s \mid b) \approx \frac{Count(s,b)}{Count(b)}$

3. Prune any pair (s,b) with $Count(s,b)$ less than a threshold T_{count} or $P(s \mid b)$ less than a threshold T_{prob}. Probabilities of the remaining pairs are renormalized.

4. Compute the new output distribution of state b. This is a sum of all the mixtures of states s remained after pruning, with $P(s \mid b)$ used to compute the new mixture weights :

$$P'(o|b) = \sum_{s:P(s|b)>0} P(s|b) \sum_{i=1}^{N_s} w_{i,s} \mathcal{N}(o; \mu_i, \Sigma_i) \quad (1)$$

$P'(o \mid b)$ is the new output distribution of state b, and N_s, $w_{i,s}$ and $\mathcal{N}(o; \mu_i, \Sigma_i)$ are the number of mixtures, the i-th mixture weight and the i-th distribution of state s in the original system, respectively. Note that as state-to-state alignments are inferred from phoneme-to-phone alignments, $P(s \mid b)$ is the same for all states of the same model.

The steps above were applied to all training sentences and the modified phonetic models were used for recognition (see section 5). The number of parameters does not increase much since Gaussian mixtures are shared and. However the average number of Gaussians per state increases.

4. DYNAMIC SLPM USING PHONETIC FEATURES

4.1. Overview

With the objective of further decreasing the Word Error Rate, we propose in this section a method to dynamically modify the phonetic models. The whole process enumerated below has been called *dynamic* in the sense that sharings of Gaussian densities are processed *during recognition* and they vary from one utterance to the next, while they are still governed by a pronunciation model. The following steps are therefore respected for each *test* utterance :

1. Some phonetic features are first extracted from the input speech on a frame-by-frame basis by a neural network.

2. At the same time, a baseline HMM system is used to apply a first recognition pass on the same input speech and to generate a lattice of the most likely word hypotheses with their time boundaries.

3. For each hypothesis, a procedure (explained later in this section) maps the word to a graph of phonetic features.

4. The graph in step 3 is compared to the phonetic features returned by the neural network in step 1 over the given word's time interval. Depending on how much the features differ (explained later in this section), some Gaussian mixtures are eventually shared between HMM models.

5. The new HMM models are used instead of the original ones for a second pass recognition.

The steps above are depicted in Figure 2 and will be explained thoroughly in the next subsections.

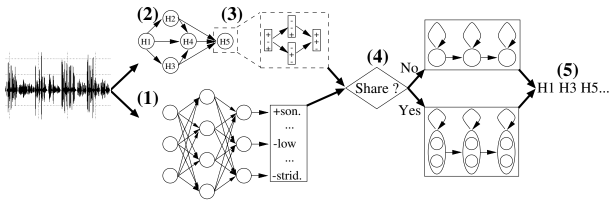

Fig. 2. Dynamic SLPM

4.2. Extraction of phonetic features

In this work we decided to rely on phonetic features to measure how far a pronunciation differs from its baseform. Phonetic features are more fundamental units than phones and cover the most relevant and controllable characteristics in speech. Our motivation in this direction was influenced by papers (e.g. [3]) mentioning that use of phonetic features provides a simple framework to understand and capture pronunciation phenomena in speech. Moreover, it was shown through experiments ([4]) that accurate classification of phonetic features is one of the most important factors to obtain better recognition performance in spontaneous speech.

Among the different existing classes of phonetic features, the SPE system (Sound Pattern of English [5]) kept our attention because of its popularity and the possibility of its representation in binary forms. A single neural network was trained to map a set of acoustic parameters presented at the inputs to a group of phonetic features (one output node per feature). The mapping is done on a frame-by-frame basis. Features are considered as independent, so that the network performs a M-from-N classification and therefore several output nodes may be activated simultaneously.

4.3. First recognition pass

A baseline HMM system (described in 5.2) applies a first recognition pass on the input speech and generates a lattice of the most likely word hypotheses, in order to reduce the search space. Consequently, only phonetic models related to these words may be subject to Gaussian sharings. Start and end times of each word are retained for later steps. To reduce computation time, if a word is present at several places in the lattice, the time interval of the hypothesis with the highest acoustic likelihood is selected.

4.4. From word hypothesis to phonetic features

Each word hypothesis must be mapped to a graph of phonetic features, which will later help to decide whether models representing this word should be transformed or not, and if so which ones. The graph is constructed thanks to the following steps (an example is given for the word "had" in Figure 3) :

1. The word's canonical pronunciation is extracted from the lexicon.

2. A forced alignment is applied on the transcription over the given word's time interval in order to get segmentation points of its phoneme constituents.

3. Each phoneme is mapped to a set of phones. The allowed set for a given phoneme is the one computed in section 3. A graph of possible phones is therefore generated. To simplify the process, phones are assumed to share the same segmentation points as their corresponding phoneme.

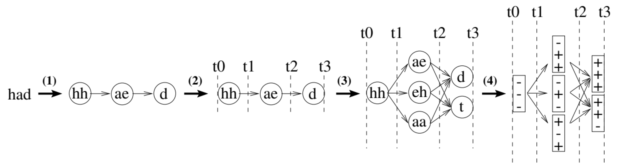

Fig. 3. Procedure to map a word to a graph of phonetic features

4. Map each phone in the graph to its corresponding vector of phonetic features using a lookup table. The vector is duplicated (not represented in Fig. 3) as many times as there are frames attributed to this phone.

4.5. Comparisons of phonetic features

The graph of features generated in the previous subsection is compared to the sequence of features returned by the neural network (cf. section 4.2). Comparisons are done phone-by-phone over the time intervals given by their segmentation points. Each comparison consists of evaluating a *Measure of Similarity* (MoS). Suppose that a sequence of theoretical feature vectors $F = \{\overrightarrow{f}, \overrightarrow{f}, \ldots\}$ (N times to cover N frames) associated to a phone and a sequence $G = \{\overrightarrow{g_1}, \overrightarrow{g_2}, \ldots, \overrightarrow{g_N}\}$ returned by the neural network over N (independent) frames. The MoS is evaluated as :

$$\log S(F \to G) = \frac{1}{N} \sum_{i=1}^{N} \log S(\overrightarrow{f} \to \overrightarrow{g_i}) \qquad (2)$$

Assuming phonetic features also as independent and supposing there are K features per vector, each term on right hand side of equation 2 is given by :

$$S(\overrightarrow{f} \to \overrightarrow{g_i}) = \frac{1}{K} \sum_{j=1}^{K} \left[1 - \left|targ(f^j) - act(g_i^j)\right|\right] \qquad (3)$$

where $targ(f^j)$ is the target value (0 or 1) of the j-th feature of the considering phone, while $act(g_i^j)$ is the activation value (in the range [0..1]) returned by the neural network for the j-th feature of the i-th frame.

For each phoneme (represented by a set of phones in the graph), the related phone p with the maximum $\log S(F_p \to G_p)$ is retained. Once this is done for all phonemes, a path going through each of these retained phones represents the best path $\mathcal{P}$. Its MoS is given by :

$$\log S_{max} = \frac{1}{N} \sum_{p \in \mathcal{P}} N_p \log S(F_p \to G_p) \qquad (4)$$

where N_p ($\sum_{p \in \mathcal{P}} N_p = N$) is the number of frames covering the phone p. Gaussian sharings are allowed for a word only if the MoS S_{max} is above a threshold T_{match} (fixed to 0.5 in experiments). Moreover, a phoneme's model may share Gaussians of an alternate distinct phone only if the MoS of this phone is higher than both the threshold T_{match} and the self MoS of the phoneme (i.e. the MoS of a phoneme to be mapped to itself). By default, a phoneme is at least always mapped to itself. The output distributions follow the same shape as in equation 1 (cf. section 3), except that sharings are further limited to states whose phones match these two conditions.

As an example, suppose that a phoneme “aa” may be mapped to the following phones with their MoS (cf. equation 2) : “aa”(0.26), “ah”(0.13), “ao”(0.40) and “ax”(0.64). The phone “ah” is not a candidate for sharing because its MoS (0.13) is both lower than the self MoS (“aa”, 0.26) and T_{match} (0.50). “ao” is not a good candidate either because its value is still lower than T_{match}. Only “ax” satisfy both conditions (0.64 $>$ 0.26 and 0.64 $>$ 0.50) and shares its Gaussian mixtures with “aa”. Probabilities of associations $P(s \mid b)$ used to compute the new mixture weights in equation 1 are therefore ($i = 1, 2, 3$) :
$P(State_i(aa) \mid State_i(aa)) = 0.26/(0.26 + 0.64) \cong 0.29$
$P(State_i(ax) \mid State_i(aa)) = 0.64/(0.26 + 0.64) \cong 0.71$

4.6. Second recognition pass

Once all hypotheses are processed and the appropriate transformations done, the new HMM models are used instead of the original ones for a second recognition pass. The lexicon is also updated to take account of changes. Note that two identical phonemes found in the new lexicon may now refer to different models (for example, “bar $\to$ b aa1 r” and “car $\to$ k aa2 r” refers to 2 different models of the phoneme “aa”).

5. EXPERIMENTS

5.1. Database and tools

All experiments were carried out on the TIMIT database [6]. All training sentences except SA files were used, as well as the core test set for evaluation. The HMM system used to build the baseline recognizer and to evaluate the different lexicons is HTK [7]. The neural network used to map from acoustic vectors to phonetic features is the NICO toolkit [8].

5.2. Baseline system

The baseline HMM system was built using the set of 40 phones proposed in [9] in order to associate each phone to a unique set of SPE features used in the experiments. A “silence” and a “short pause” model were added to it. All models have three “left-to-right” states (no skips), except for “short pause” that has only one state tied to the center state of “silence” and for which skip of the model is allowed. The system was trained using 39 MFCC coefficients (12 static + 1 energy, 13 Δ, 13 $\Delta\Delta$) and the hand transcriptions of TIMIT. The resulting models are monophones with 10 Gaussian mixtures per state. Evaluated on the core test set of TIMIT, the system achieved a 14.8% Word Error Rate (WER) (85.2% accuracy).

5.3. Recognition results on phonetic features

For sake of compatibility with the trained HMM system, the neural network was also trained using the same set of phones found in [9] as well as their corresponding sets of SPE features (14 + 1 silence). The chosen topology and training method are almost identical to those reported in [10]. Comparisons between recognized features and those derived from the hand phone transcriptions of TIMIT led to the results in Table 1, given in percentage of frames correct on the cross-validation set.

The results show that each feature taken separately can be reliably recognized. The ”all correct” shows how frequently all features are simultaneously correct for a given frame. We see that

Feature	Correct (%)	Feature	Correct (%)
sonorant	95	round	92
syllabic	90	anterior	90
consonantal	91	coronal	88
high	88	voice	89
back	92	continuant	91
front	93	nasal	98
low	92	strident	97
		silence	98
Average	**92**	**All correct**	**53**

Table 1. Frame recognition results on SPE features

in average more than one frame out of two is phonetically well-identified, reminded that feature outputs are set as independent and so $2^{15} - 1 = 32767$ combinations lead to an error.

5.4. Results using the static SLPM

The sequence of steps described in section 3 was applied to the training sentences of TIMIT. At the end of the procedure, we obtained all possible pairs of states (s,b) with their probabilities of alignments $P(s \mid b)$. Instead of fixing thresholds T_{count}, T_{prob} (used to prune unreliable pairs), we preferred to set T_{count} to zero (no "count" threshold), and let the probability threshold T_{prob} be variable, in order to see the evolution of WER with respect to the number of sharings. Namely, sharings were progressively added one after another, starting with the pair with the highest probability of association. The graphic in Figure 4 shows the results obtained for the 20 first sharings. We notice that the new Word Error Rates constantly vary around the baseline WER. Another experiment showed the same kind of behaviour when each sharing is applied separately, one at a time. From the results, it seems that any improvement brought by sharing Gaussian densities for the correct words in the test set may be counterbalanced by its influence to the wrong words as well. The best WER obtained among the 20 first sharings is 14.4% WER (85.6% accuracy). We did not get any more improvement by further increasing the number of sharings.

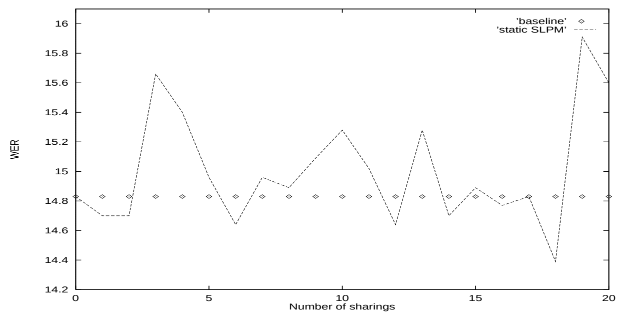

Fig. 4. Evolution of the WER in static SLPM

5.5. Results using the dynamic SLPM

HTK uses the Token Passing Model [7] to perform a N-best recognition and to generate a lattice of word hypotheses. 3 tokens in each state were used in these experiments. Thresholds T_{count}, T_{prob} and T_{match} were fixed to 10, 0.01 and 0.5 respectively. Short words (such as "a", "or", ...) were excluded from sharings because the corresponding time intervals returned by the HMM system at the first pass were often wrong. Results are given in Table 2.

Lexicon	WER (%)
Baseline	14.8
Static SLPM	14.4
Dynamic SLPM	14.0

Table 2. Recognition results with static and dynamic SLPM

Results show that a dynamic approach helps to further improve performance. Namely, we achieved a 5.4% relative reduction in WER compared to the baseline system and 2.7% compared to the static method.

6. CONCLUSION

We showed in this paper that by dynamically sharing Gaussian densities across phonetic models, it was possible to make the system more robust to pronunciation variability. A dynamic approach, based on comparisons of phonetic features, seems to better adapt the system to speech than a static method. Future experiments will also include application of this method to spontaneous speech.

7. REFERENCES

[1] H. Strik and C. Cucchiarini, "Modeling pronunciation variation for ASR : a survey of the literature", Speech Communication, Vol. 29, Nos 2–4, pp. 225–246, 1999.

[2] M. Saraçlar, H. Nock and S. Khudanpur, "Pronunciation modeling by sharing Gaussian densities across phonetic models", Computer Speech and Language, Vol. 14, No 2, pp. 137–160, 2000.

[3] K. Stevens, "Applying phonetic knowledge to lexical access", Proc. Eurospeech-95, pp. 3–11, 1995.

[4] S. Greenberg and S. Chang, "Linguistic dissection of Switchboard-corpus automatic speech recognition systems", ISCA ITRW ASR-2000, pp. 195–202, 2000.

[5] N. Chomsky and M. Halle, "The sound pattern of English", MIT Press, Cambridge, 1968.

[6] L. Lamel, R. Kassel and S. Seneff, "Speech database development : design and analysis of the acoustic-phonetic corpus", DARPA Speech Recognition Workshop, pp. 100–109, 1986.

[7] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev and P. Woodland, "The HTK Book, Version 2.2", Cambridge University Engineering Department, 1999.

[8] N. Ström, "The NICO toolkit for artificial neural networks", http://www.speech.kth.se/NICO, 1996.

[9] T. Brondsted, "A SPE based distinctive feature composition of the CMU label set in the TIMIT database", Technical Report IR 98-1001, Center for PersonKommunikation, Institute of Electronic Systems, Aalborg University, 1998.

[10] S. King and P. Taylor, "Detection of phonological features in continuous speech using neural networks", Computer Speech and Language, Vol. 14, No 4, pp. 333–353, 2000.