IDIAP RESEARCH REPORT

Hierarchical approach for spotting keywords

Mikko Lehtonen [1] [2] Petr Fousek [1] [3]
Hynek Hermansky [1]

IDIAP–RR 05-41

July 2005

submitted for publication

[1] IDIAP Research Institute, Martigny, Switzerland
[2] Helsinki University of Technology, Helsinki, Finland
[3] Czech Technical University in Prague, Prague, Czech Republic

IDIAP Research Institute **www.idiap.ch**
Rue du Simplon 4 P.O. Box 592 1920 Martigny – Switzerland
Tel: +41 27 721 77 11 Fax: +41 27 721 77 12 Email: info@idiap.ch

IDIAP Research Report 05-41

Hierarchical approach for spotting keywords

Mikko Lehtonen Petr Fousek Hynek Hermansky

July 2005

submitted for publication

Abstract. The paper presents a new approach to spotting a particular sound (keyword) in an acoustic stream. The approach is based on hierarchical processing where equally-sampled posterior probabilities of phoneme classes are estimated first, followed by matched filtering that yields non-equally spaced values, one for each phoneme, indicating confidences of underlying phoneme being present. The target keyword is indicated by a particular sequence of high-confidence phonemes.

1 Introduction

Daily experience suggests that not all words in the conversation, but only a few important ones, need to be accurately recognized for satisfactory speech communication among human beings. The important *key-words* are more likely to be rare-occurring high-information-valued words. Human listeners can identify such words in the conversation and possibly devote extra effort to their decoding. On the other hand, in a typical automatic speech recognition (ASR) system, acoustics of frequent words are likely to be better estimated in the training phase and language model is also likely to substitute rare words by frequent ones. As a consequence, important rare words are less likely to be well recognized. Keyword spotting by-passes this problem by attempting to find and recognize only certain words in the utterance while ignoring the rest. Doing this in a confident way would open new possibilities in human-computer interaction.

With errorless large vocabulary continuous speech recognition (LVCSR), keyword spotting would be trivial. However, the first step in a typical keyword spotting system is usually still a conventional Hidden Markov Model based ASR, where the keywords are represented by usual word-models and all other words are represented by an ergodic *garbage* model. The keyword spotting performance increases with increasing the ability of the *garbage* model to represent the non-keyword speech [1] [2].

1.1 Proposed approach

Since keyword spotting is relatively younger than ASR, it is not clear if the LVCSR-based keyword spotting approaches are a consequence of a simple inertia in engineering where any new problem is seen in the terms of the old one, or the optimal strategy. In this work we study an alternative approach where the goal is to find the target sounds from an acoustic stream while ignoring the rest. These target sounds are sub-units of the keyword so a correct sequence of target sounds indicates the keyword. When the key-sounds are words, we can do this by looking at the phonemes of the keyword while ignoring the other phonemes.

This goal is achieved by hierarchical processing where first equally-spaced posterior probabilities of phoneme classes are derived from the signal, followed by matched-filter processing that selectively discards less informative parts of the utterance to derive a sequence of underlying phonemes. Subsequently, the target keyword is indicated by the appropriate sequence of the derived phonemes.

The paper is organized as follows. Section 2 presents the three steps of the hierarchical processing we use for keyword spotting. The experiments and results are presented in Section 3 and we finish with discussion and conclusions in Section 4.

2 Steps of hierarchical processing

2.1 From acoustic stream to phoneme posteriors

The first step derives estimates of phoneme posteriors at 10 ms steps (100 Hz sampling rate) from the speech data. This is accomplished as follows: First a critical-band spectral analysis (auditory spectral analysis step from PLP technique [3]) is carried out and a bank of 2-D bandpass filters with varying temporal and spectral resolutions is applied to the resulting critical-band spectrogram. We implemented the 2-D filtering by first processing a trajectory of each critical band with temporal filters and subsequently applied frequency filters to the result. By filtering temporal trajectory of each critical band with a bank of fixed length low-pass FIR filters representing Gaussian functions of several different widths (determined by standard deviation σ) and by subsequently computing first and second differentials of the smoothed trajectories we would obtain a set of modified spectra at every time step. The same filter-bank is applied to all bands.

In the implementation, we use directly the discrete versions of the first and second analytic derivatives of a Gaussian function as impulse responses. Filters with low σ values yield finer temporal resolution, high σ filters cover wider temporal context and yield smoother trajectories. All temporal

filters are zero-phase FIR filters, i.e. they are centered around the frame being processed. Length of all filters is fixed at 101 frames, corresponding to roughly 1000 ms of signal, thus introducing a processing delay of 500 ms. First and second derivatives of Gaussian function have zero-mean by the definition. By using such impulse responses we gain an implicit mean normalization of the features within a temporal region proportional to the σ value, which infers robustness to linear distortions. Since we use discrete impulse responses with a length fixed at 101 samples, we approximate real Gaussian derivatives with certain error, which increases towards both extremes of σ, thus limiting the possible σ values to a range of approximately 6–130 ms. In our experiments we use eight logarithmically spaced impulse responses in a σ range 8–130 ms. These responses representing the first and the second Gaussian derivatives are shown in the left and right parts of Figure 1, respectively. Related frequency responses are illustrated in Figure 2.

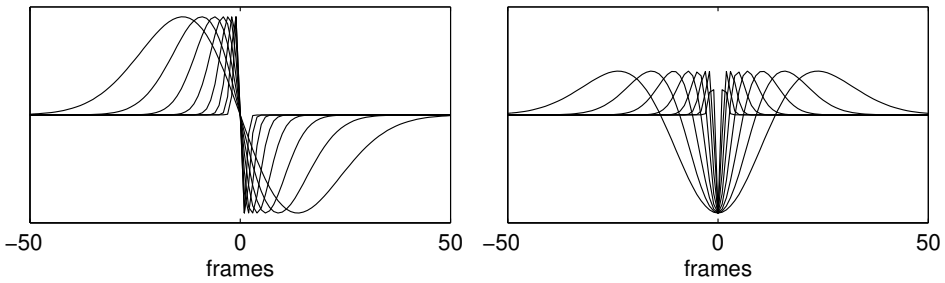

Figure 1: Normalized impulse responses of the two sampled and truncated Gaussian derivatives for standard deviations $\sigma = 8 - 130$ ms.

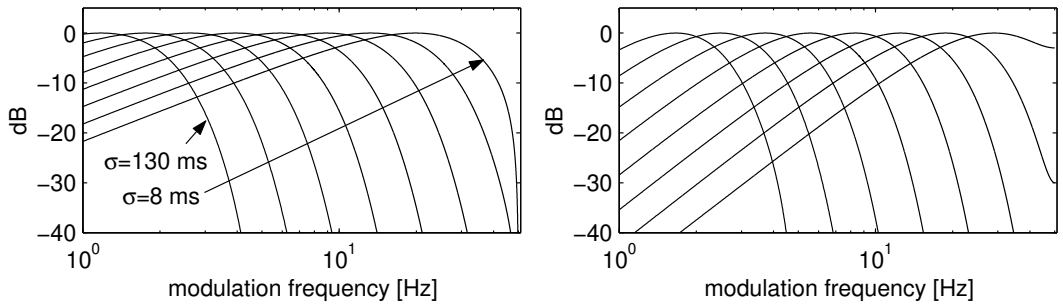

Figure 2: Normalized frequency responses of first two sampled and truncated Gaussian derivatives for standard deviations $\sigma = 8 - 130$ ms.

Temporal filtering of 15 critical band trajectories with a bank of 2×8 filters (two derivatives of the Gaussian at eight different standard deviations) results in 16 modified auditory spectra at every 10 ms step, containing overall $15 \times 2 \times 8 = 240$ features.

Subsequently we implement the full 2-D filtering by applying frequency filters to the modified auditory spectra. The first frequency derivative is approximated by a 3-tap FIR filter with impulse response {-1.0; 0.0; 1.0}, introducing three-bands frequency context. This time-invariant filter is applied across frequency to each of the 16 modified auditory spectra. Since derivatives for the first and last critical bands are not defined, we obtain $(15 - 2) \times 16 = 208$ features.

Final feature vector is formed by concatenating the 240 features from temporal filtering with the 208 features from the full 2-D filtering, thus yielding 448 features. This feature vector is fed to an MLP neural net classifier (TANDEM probability estimator [4]), which is trained to give an estimate of phoneme posterior probabilities at every 10 ms step. An example of trajectory of such phoneme posterior probability estimates (posteriogram) is shown in Figure 3.

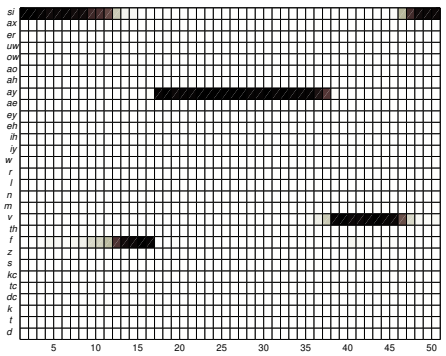

Figure 3: Posteriogram of the word *five* followed by silence.

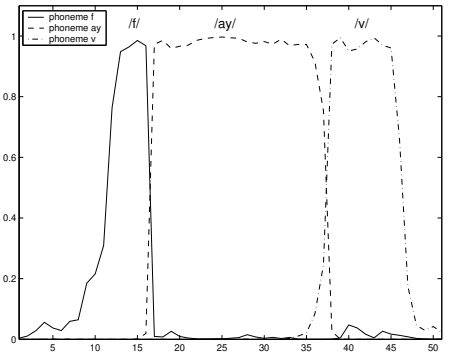

Figure 4: Trajectories of phoneme probability estimates.

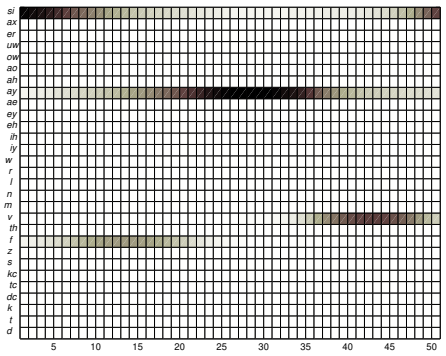

Figure 5: Posteriogram of Figure 3 after filtering with the bank of matched filters of Figure 15 and normalization.

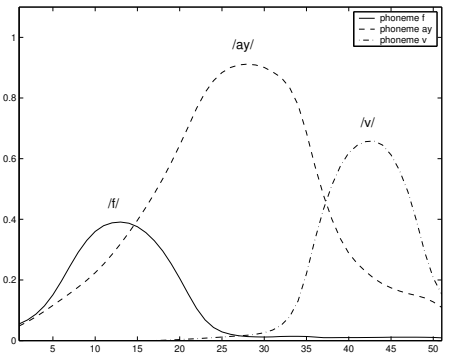

Figure 6: Filtered and normalized trajectories of phoneme probability estimates.

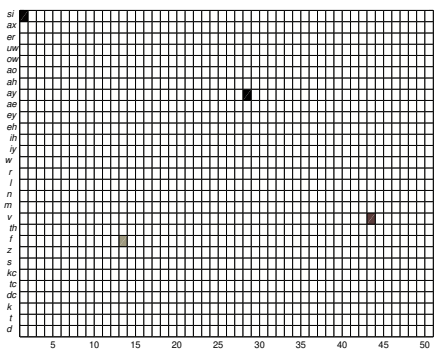

Figure 7: Phoneme-spaced posteriors of the utterance of Figure 3 (filtered posteriogram after peak extraction).

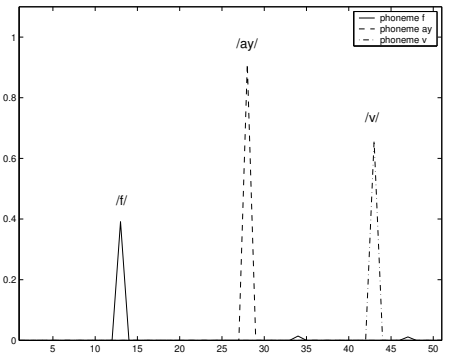

Figure 8: Phoneme-spaced posteriors of the phonemes of word *five*.

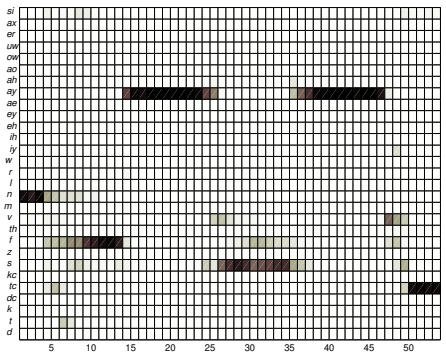

Figure 9: Difficult case: Posteriogram of utterance *five five* with classification error and weak posteriors.

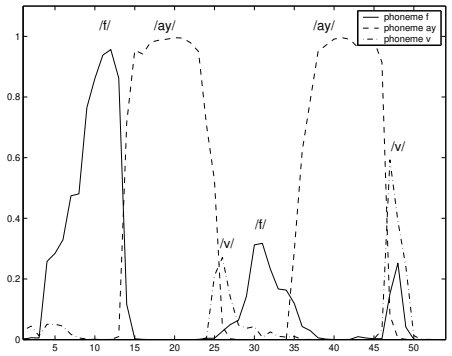

Figure 10: Trajectories of phoneme probability estimates for word *five*.

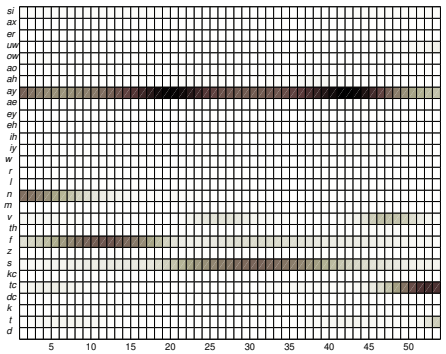

Figure 11: Posteriogram of Figure 9 after filtering with the bank of matched filters of Figure 15 and normalization.

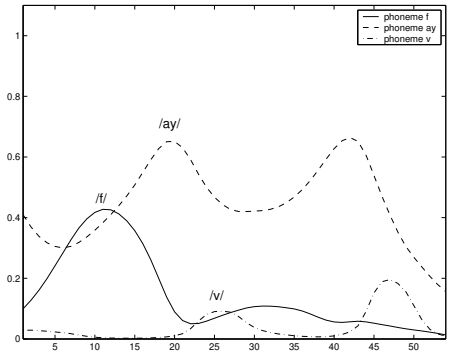

Figure 12: Filtered and normalized trajectories of phoneme probability estimates.

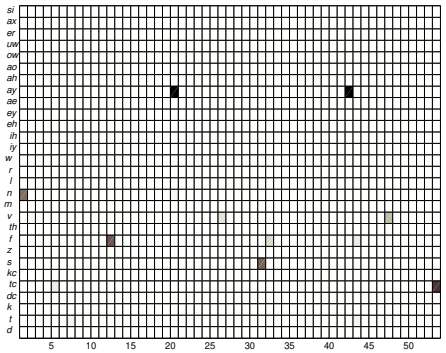

Figure 13: Phoneme-spaced posteriors of the utterance of Figure 9.

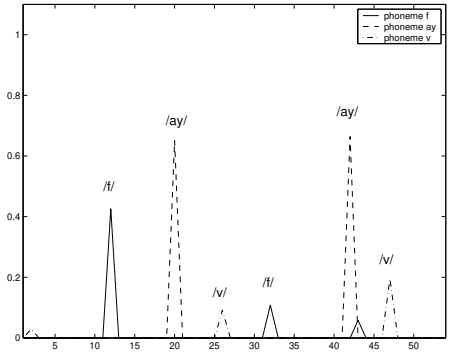

Figure 14: Phoneme-spaced posteriors of the phonemes of word *five*.

2.2 From frame-based phoneme posteriors to phoneme-spaced posteriors

Even though (as illustrated in the Figure 3) to human eye the frame-based posterior estimates often clearly indicate the underlying phoneme sequence, the step from the frame-based estimates to phoneme-level estimates is very important. It involves nontrivial operation of information rate reduction (carried sub-consciously by human visual perception while studying the posteriogram) where the equally sampled estimates at the 100 Hz sampling rate are to be reduced to non-equally sampled estimates at the phoneme rate of about 15 estimates/s. In the conventional (HMM-based) system, this is accomplished by searching for an appropriate underlying sequence of hidden states.

We have opted for more direct communication-oriented approach where we postulated existence of bank of matched filters for temporal trajectories of phoneme posteriors, one filter per phoneme, with impulse responses derived by averaging 0.5 s long segments of trajectories of the respective phonemes, aligned at the phoneme centers.

In deriving these averages, we need to deal with cases where the center phoneme of the 0.5 s segment is repeated after one phoneme in the transcription and segments where phoneme center is less than 0.25 s from the beginning or end of the utterance. In the current work, these segments were not included in the average.

Resulting set of filters is shown in Figure 15, and an example of the posteriogram from Figure 3 filtered by this bank of matched filters is illustrated in Figure 5.

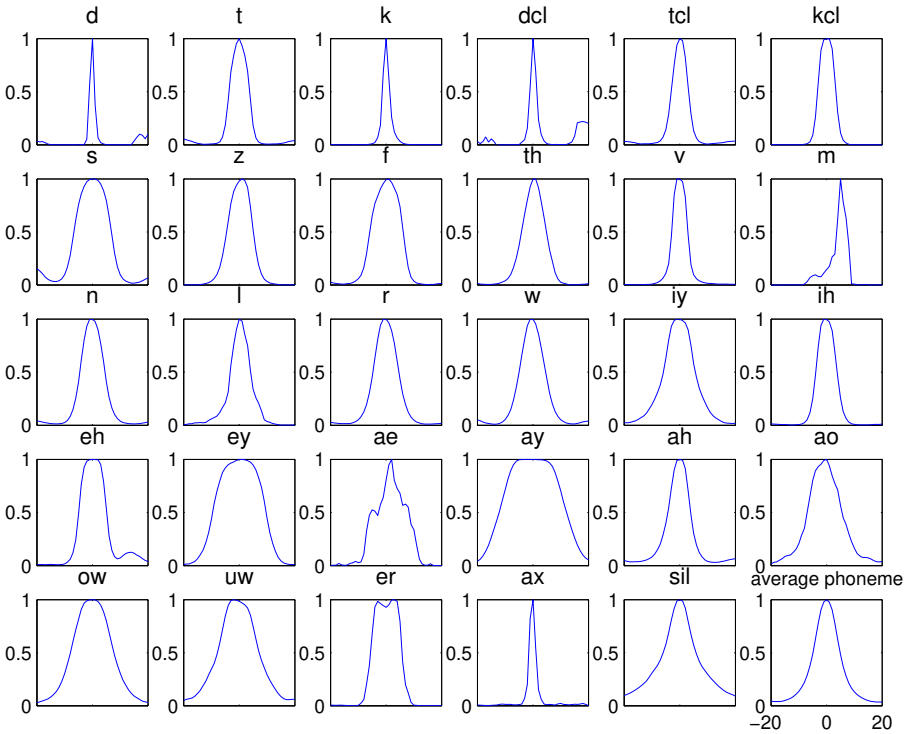

Figure 15: The matched filter bank of mean temporal trajectories of phoneme posteriors, estimated from 30,000 phonemes of the training data. The non-zero forms outside the main peak are due to the small vocabulary, where regular phoneme sequences affect the average temporal trajectories (e.g. in digits, *s* is most often in word *six*). The filter lengths are 41 frames (410 ms) and the maxima are scaled to one for the illustration. The impulse responses of phonemes who are very rare in digits (e.g. *m* and *ae*) are not well estimated.

In the next step, local maxima (peaks) for each filtered trajectory were found. The values in the peaks were taken as estimates of probability that the center of the given phoneme is aligned with the center of the impulse response of the respective matched filter and retained, all other data were discarded. An example of such re-sampled posteriogram is shown in Figure 7.

In an alternative approach, only one universal filter was derived by averaging impulse responses of all phoneme-specific filters. Results with this alternative single filter were similar but not better than the results of the phoneme-specific filters.

2.3 Filtering issues

When all the bands of the MLP output are filtered with the same filter, the sum of the phoneme posterior estimates remains one in every frame. When N_p is the number of phonemes in the MLP output, $\tilde{P}(P_t = j|x_t)$ is the posterior probability of phoneme j at time t after filtering, $L_{1/2} = \frac{L-1}{2}$, where L is the length of the filter, and w_i is the coefficient i of the filter's impulse response, the frame wise sum of filtered posteriors yields:

$$\begin{aligned}\sum_{j=1}^{N_p} \tilde{P}(P_t = j|x_t) &= \sum_{j=1}^{N_p} \sum_{i=-L_{1/2}}^{L_{1/2}} w_i P(P_i = j|x_i) \\ &= \sum_{i=-L_{1/2}}^{L_{1/2}} w_i \sum_{j=1}^{N_p} P(P_i = j|x_i) = \sum_{i=-L_{1/2}}^{L_{1/2}} w_i = 1\end{aligned} \tag{1}$$

However, we use a filter bank, so the filter coefficient of a frame i depends also on the band j (w_i becomes $w_{i,j}$), and the value of the sum is no longer guaranteed to be one. Therefore the filtered posterior estimates are normalized in every frame so that they add to one.

2.4 From phoneme-spaced posteriors to words

To move from phoneme-based posteriors to words we look for consecutive peaks of the keyword's phonemes. We defined a minimum and a maximum duration for each interval between two consecutive phonemes and if the successive phoneme of a keyword is not found within these limits, the alarm is not set. We also defined a minimum and a maximum duration between the first and last peaks of the keyword.

The minimum and maximum durations of these intervals are defined from keywords in the training data. The limits for all intervals are individually chosen in such a way that 99.5% of keywords in the training data fit inside the interval. Figure 16 illustrates how these intervals are estimated for the keyword *one*.

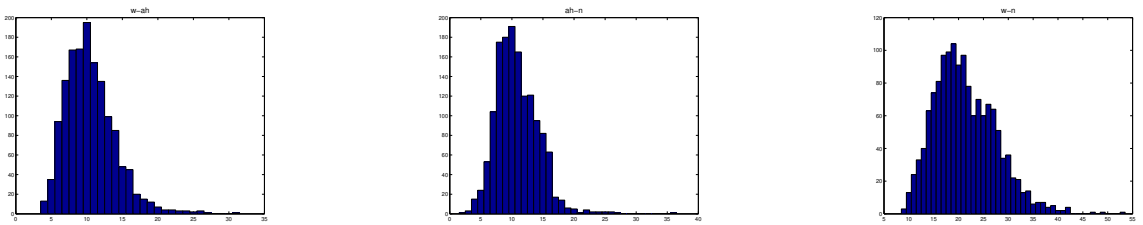

Figure 16: Histograms of the distances (in frames) between phonemes *w* and *ah*, *ah* and *n*, and *w* and *n* from words *one* of the training data. Corresponding 99.5% intervals are 4-26 frames (*w*-*ah*), 3-26 frames (*ah*-*n*) and 10-40 frames (whole word).

All the local maxima in the filtered posteriogram do not correspond to phonemes. Nevertheless, the higher the peak is, the higher is the probability of underlying phoneme. Therefore peaks below certain thresholded are eliminated. A single threshold is applied for all phonemes. This thresholds is one parameter of our system that determines the trade off between detection rate and false alarm rate.

In cases where the first phoneme has two local maxima before the second phoneme, only later one of these maxima is taken into account. Also if one of the other phonemes is repeated before the successive phoneme, the higher one is accepted.

2.5 Example: Difficult case

Example keyword *five* of Figure 3 is easy to spot because the phoneme segments are well classified in the posteriogram. However, this is not always the case. Figures 9 and 10 illustrate a more difficult case for the same word, where the speech rate is higher and there are classification errors.

Figures 11 and 12 show what is the effect of filtering for these weaker and shorter trajectories of posterior estimates. Figures 13 and 14 illustrate that the phonemes of the keyword remain present in the phoneme-spaced posterior estimates, though the weakest of these phoneme estimates are in the level of false alarm peaks (peaks that do not correspond underlying phonemes).

3 Experiments

3.1 The setup

Proposed keyword spotting algorithm was tested on a small vocabulary continuous speech task. Two telephone speech corpora were used: OGI-Stories containing spontaneous speech [5] and OGI-Numbers95 providing continuously pronounced numbers [6]. The MLP neural network estimating phoneme posteriors of 29 English phonemes was trained on a joint set of both corpora, namely the phoneme-labeled part of OGI-Stories ($\sim$2.7 hrs) and the training part of OGI-Numbers95 ($\sim$1.7 hrs). Approximately 10% of the data was used for MLP cross-validation. Trained MLP was subsequently used to give an estimate of posteriors of a data sets used for training matched filters and for testing. Matched filters were derived on a subset of the training part of OGI-Numbers95 containing only utterances with strings of digits (from *zero* to *nine* plus *oh*), overall $\sim$1.3 hrs. Test data set consisted of the testing part of OGI-Numbers, also containing only utterances with digits strings ($\sim$1.7 hrs).

We implemented separate keyword spotters for all the digits using the primary pronunciation form of each digit as the keyword model and handled all the pronunciation variants as keywords (expect for keyword *zero* the single phoneme form *ow* was not used). The number of keywords that are not found due to this limitation is not the same for all digits. Searching all the legal pronunciation variants would yield better results, but this issue is not addressed in the current work. In some cases there were more than one hits from one keyword. Because these hits appear partly at the same time, they were automatically merged.

3.2 Results

The performance of each keyword spotter is evaluated by comparing the keyword detection probability (estimated by the detection rate) to the number of false alarms per one hour of speech. Table 1 presents the detection rates for all the ten digits at false alarm rates 5 and 10 false alarms per hour. For keywords that yield the best results the detection rates are almost the same with both false alarm rates. In our experiments the keyword detection rates keep improving until false alarm rate of about 20 per hour.

The differences in performance for different keywords are large, worst results were obtained for the word *eight*. This is because some phonemes are more prone to classification errors than others, the probability that a digit is mixed with another one (false alarm) is not the same for all digits,

Table 1: Keyword spotting probabilities for all single digit keyword spotters at false alarm (FA) rates 5 and 10 per hour.

Keyword	Detection rate 5 FAs/h	Detection rate 10 FAs/h
One	93.4%	95.2%
Two	85.0%	90.9%
Three	95.0%	96.1%
Four	90.4%	92.5%
Five	82.9%	90.5%
Six	94.3%	95.2%
Seven	91.0%	91.6%
Eight	41.6%	51.6%
Nine	46.7%	69.1%
Zero	90.6%	92.5%

the effect of threshold is not the same for all phonemes and the differences between real and target pronunciation vary among keywords.

Keyword *seven* had five legal pronunciations in our experiments, but searching only one of them still yielded 91.6% recognition rate for 10 false alarms per hour. In this case, however, most alternative pronunciations the keyword appeared as the most common canonical pronunciation in the posteriogram.

In other words the MLP made often classification errors with similar phonemes like *eh* and *ah*. Thus, *nine* was often misrecognized because the phoneme *ah* of *one* was mixed with phoneme *ay*, which caused false alarms for keyword *nine*.

We examined further the effect of different pronunciations of *eight*. To do that, we also applied the keyword spotter for spotting the the close form *ey-tcl* in addition to the target pronunciations *ey-t* and merged the results. This yielded an accuracy of 66,2% in false alarm rate 10 per hour. In another test, we merged the posterior estimates of phonemes *tcl* and *t* to form a new target phoneme for the keyword. This method yielded detection rate of 78,9% in false alarm rate 10 per hour. However, the performance for *eight* remained relatively low compared to other digits. Still, two thirds of the remaining alarms involved another closure phoneme (*kcl*) or silence classified as *tcl*, which suggests that the performance for *eight* was degraded by classification errors of closures. These small tests clearly indicate further potential for improvements of the proposed approach.

4 Discussion and Conclusions

We have presented an approach to keyword spotting that does not use conventional temporal processing techniques such as HMMs or dynamic time warping (DTW). Our technique looks only for the target sounds in the particular keyword and ignores the rest. This approach was applied quite successfully for spotting a particular digit in the stream of other digits.

The performance in our experiments depends mostly on the frame-level phoneme posterior estimates. The matched filtering we used seems feasible for finding the phonemes from the posteriogram. Recognition errors with similar phonemes helped finding different pronunciations variants of the keyword in some cases (e.g. *seven*), but increased the false alarm rate in other cases (e.g. *nine*).

In this particular study of spotting digits, the step from phoneme-spaced posteriors to words is not complex, because no phoneme streams of a keyword are found in another word. In general, this may of course happen and has to be addressed to avoid false alarms. To apply it in the general case of unconstrained speech, one would have to deal with issues of finding words from more complex

phoneme sequences and possibly with less reliable phoneme estimates. Thus, it is yet to be seen how this approach will perform with unrestricted speech.

5 Acknowledgements

This work was partly supported by the European Union 6th FWP IST Integrated Project AMI (Augmented Multi-party Interaction, FP6-506811, publication).

References

[1] Rose, R., Paul, D.: A Hidden Markov Model Based Keyword Recognition System. In Proceedings of ICASSP '90, pp. 129–132, Albuquerque, New Mexico, United States, 1990.

[2] Rose, R.: Keyword Detection In Conversational Speech Utterances Using Hidden Markov Model Based Continues Speech Recondition. Computer Speech and Language, 9, 309–333, 1995.

[3] Hermansky, H.: Perceptual linear predictive (PLP) analysis of speech, J. Acoust. Soc. Am., Vol. 87, No. 4, April 1990.

[4] Hermansky, H. et al: Connectionist Feature Extraction for Conventional HMM Systems. In Proceedings of ICASSP '00, Istanbul, Turkey, 2000.

[5] Cole R. et al.: Telephone Speech Corpus Development at CSLU. In Proceedings of ISCLP '94, pp. 1815–1818, Yokohama, Japan, 1994.

[6] Cole R. et al.: New Telephone Speech Corpora at CSLU. In Proceedings of Eurospeech '95, pp. 821–824, Madrid, Spain, 1995.