ACOUSTIC KEYWORD SPOTTER – OPTIMIZATION FROM END-USER PERSPECTIVE

Igor Szöke, F. Grézl, J. "Honza" Černocký, M. Fapšo*

Brno University of Technology, Speech@FIT
Brno, Czech Republic
szoke@fit.vutbr.cz

Tomáš Cipr

Phonexia
Brno, Czech Republic
cipr@phonexia.com

ABSTRACT

The paper deals with the development of acoustic keyword spotter (KWS) meeting requirements of a real user from the security community. While the basic scheme of the KWS is relatively standard, it uses novel features derived by a hierarchy of neural networks, and score normalization trained to maximize a user-like evaluation metric. The results are reported on a selection of Czech conversational telephone speech (CTS), radio and read data.

Index Terms— keyword spotting, spoken term detection, neural networks, calibration

1. INTRODUCTION

The task of searching keywords or terms in spoken data has been studied for several years and different methods, ranging from simple acoustic keyword spotting to complex spoken term detection based on LVCSR, have been developed. This paper describes our efforts to develop accurate, robust and fast keyword spotting for Czech language. The work was done in tight cooperation with Czech Ministry of Interior, that had the following user requirements:

- speed (faster than real-time)
- real telephone conversational speech
- channel and noise robustness
- processing of continuous stream of data (no segmentation into different calls)
- one executable

An important aspect was also the framework of deployment of the keyword spotter: it should operate in "process once – search once" mode, so that we have discarded any indexing/search schemes. The set of keywords is relatively stable, contains up to 100 words of interest, and may change several times per year. The word spotter should also work in "filtering" mode: The amount of incoming data is much larger than what can be processed by human operators. That is why it is important to select only recordings having occurrences of the keywords with high confidences.

According to the requirements listed above, we decided to use an acoustic keyword spotter based on phone posteriors estimated by artificial neural network [1] . In comparison to our previous work, we aimed to do better speaker and channel adaptation, use more training data, improve keyword score calibration and evaluate channel robustness.

2. SYSTEM DESCRIPTION

This section deals with description of keyword spotting system architecture, feature extraction, artificial neural network topology and calibration of keywords. The overall scheme of our acoustic keyword spotting system is in Figure 1. After the VAD, input speech signal is converted to a stream of features. These features are processed by artificial neural network classifier, producing 3-state phone posterior probabilities. These posteriors, after application of logarithm, are processed by a keyword spotting decoder that produces putative hits of the keywords. Keyword scores are then calibrated according to phones the keyword consists of. Finally, threshold is applied to filter out detections having low confidence.

2.1. Voice activity detection

At the beginning, two-step voice activity detection (VAD) is performed. The first step is based on simple set of heuristics applied on spectrum, energy and signal. This VAD filters out silence or technical noises (beeps or faxes). Next, this "clean" signal is sent to a small 4-layer neural network (NN) (Figure 2). It has 200 neurons in hidden layers and 38 outputs representing 37 phones and 1 silence. Phones are merged to `speech` segments. The VAD neural network input features are the same as is presented in section 2.2, only vocal tract length normalization and mean/variance normalization

*This work was partly supported by Czech Ministry of Interior project No. VD20072010B16, Grant Agency of Czech Republic project No. 102/08/0707, Czech Ministry of Education project No. MSM0021630528, and by BUT FIT grant No. FIT-10-S-2. Grézl was supported by Grant Agency of Czech Republic under project No. GP102/09/P635. We thank Technical University of Liberec and University of West Bohemia for making their prompted and read telephone data available, for cooperation on data preparation (annotations, definition of sets) and for implementation of the scoring tool used.

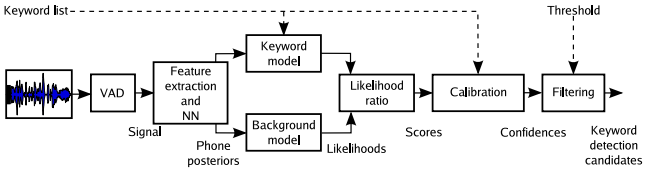

Fig. 1. Block scheme of acoustic keyword spotter.

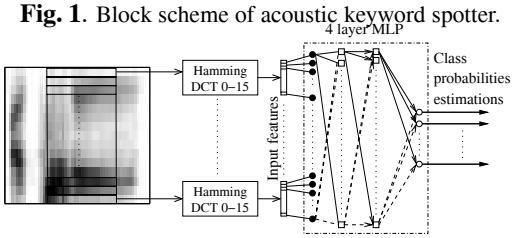

Fig. 2. Scheme of 4 layer neural network used for voice activity detector. The input are 15 critical bands in log domain (similar to figure 3). 310ms long temporal patern of each critical band is taken (31 numbers per critical band), Hamming window and DCT is applied. First 16 DCT coefficients times 15 critical bands are then concatenated and fed into the 4-layer neural net. The output of the net are 38 phone posteriors merged to two clases – *speech* and *nonspeech*.

are omitted. For the VAD NN, the length of temporal patterns is 310ms ($L_{TP} = 31$) and it is reduced by the DCT to 16 coefficients ($L_{DCT} = 16$).

2.2. Feature extraction

The system is trained and tested on telephone conversational speech (8kHz data). Figure 3 presents the feature extraction used. Input speech is first segmented into frames and power spectrum is calculated for each frame. VTLN is applied, energies from 15 Mel-scale critical bands ranging from 64Hz to 3800Hz are extracted, and passed through logarithm. Next, mean normalization is performed on segments of speech detected by the VAD enlarged by 100ms. We obtain so-called log-critical band spectrogram (CRB), from which long temporal patterns of length L_{TP} are extracted. Hamming window and dimensionality reduction by DCT to L_{DCT} coefficients are applied to each long temporal critical band trajectory. Finally, these reduced temporal patterns are concatenated to one feature vector and fed into the NN.

2.3. Generation of phone-state posteriors

The topology of the NN is crucial. Based on our previous experiments in LVCSR [2], we use a hierarchical structure called *bottle-neck universal context network* (Figure 4). It consists of two parts: a context network and a merger.

The input of context network is a context of $L_{TP} = 11$ critical-band energies around the current frame, reduced by DCT to $L_{DCT} = 6$ parameters, so that the input size is $15 \times 6 = 90$. The context NN is so-called *bottle-neck network*. It is trained as 5 layer network having the 3^{rd} layer as the bottleneck of size 80 neurons. The sizes of 2^{nd} and 4^{th} layers are

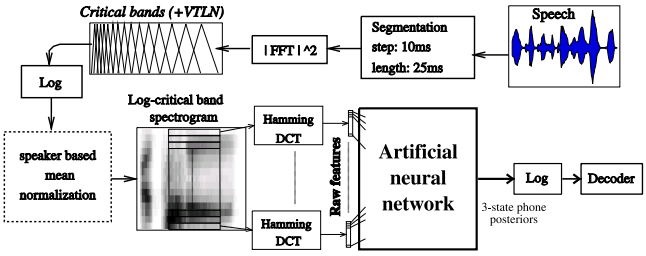

Fig. 3. Scheme of feature extraction. Signal is segmented into 25ms long frames each 10ms. Power spectrum is calculated on each frame. Then 15 Mel-critical bands with VTLN are applied and the result is converted to log domain. After segment-based mean normalization, the *Log-critical band spectogram* is source of features for neural network classifier (figure 4). The neural network produces 3-state phone posteriors, which are converted to log domain and processed by the decoder.

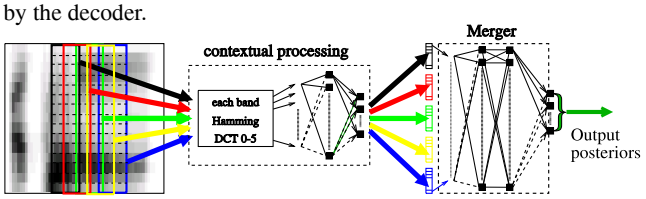

Fig. 4. Scheme of universal context (UC) neural net architecture used in figure 3. The input features – log-critical band spectogram – are split into 5 blocks of length 11 (in time – x-axis) and height 15 (critical bands – y-axis). These blocks are sampled each fifth frame, so they are overlapped by 6 frames. For each block, 110ms long temporal patern of each critical band is taken (11 numbers per critical band). Then Hamming window and DCT are applied. First 6 DCT coefficients times 15 critical bands are then concatenated and fed into the 3-layer "UC" neural net. The outputs of the "UC" neural net for each block are concatenated and fed into the "merger" (4-layer neural net).

1373 and the number of outputs corresponds to $3 \times 38 = 114$ – the number of 3-states phone posteriors[1]. The fourth and fifth layers are cut-off after the training so the output size of context network is 80 (Figure 5).

The merger receives 5 context net outputs sampled every 5 frames (for frame t, this is $t-10, t-5, t, t+5, t+10$), so that it actually "sees" a 310ms context in the CRB matrix and the merger input size is $5 \times 80 = 400$. The merger is a standard 4 layer neural NN. Its outputs are 114 phone-state posteriors. More information on the universal context can be found in [2].

2.4. Keyword spotting decoder

Phone-state posteriors produced by the universal context neural network are processed by logarithm and fed to the decoder: a standard Viterbi scheme, slightly modified to calculate likelihood – see Figure 6. The filler model (A) should model all

[1] For Czech, we use 37 phonemes plus silence.

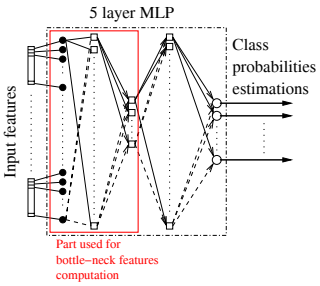

Fig. 5. Training of bottle neck neural network.

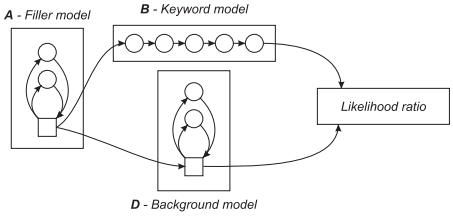

Fig. 6. Evaluation of likelihood ratio in acoustic keyword spotting.

speech preceding the keyword, it is a free phone loop. The keyword model (B) is a concatenation of phone models of which the keyword consists. The background model (D) is again a single phone loop. In the real recognition network, there is only one phone loop (Figure 7). The likelihood ratio of particular keyword is normalized (divided) by the length of the detection before leaving the decoder.

2.5. The data and evaluation

The keyword spotting system was trained on Czech data set denoted *CTS-RR*. The data contains 100.9h of speech consisting of: 45.6h of real conversational telephone speech[2], 18.9h of radio telephone speech (people calling into broadcasts) and 36.4h of read or prompted speech recorded through telephone. We also trained on smaller – only target – set denoted as *CTS* which contains only the 45.6h of real conversational telephone speech.

We defined one development set denoted as *DEV* (2.2h of real conversational telephone speech) and three evaluation sets denoted as *TEST* (2.0h of real conversational telephone speech), *Radio* (1.1h of radio telephone speech) and *SpDat* (5.3h of read speech from Czech SpeechDat-E[3]). The *DEV* set was used for tuning (penalties, score calibration, etc.).

Each of the sets had also defined set of keywords. The *DEV* keyword list consists of 569 "nice" longer discriminative keywords appearing 697 times. The *TEST* keyword list consists of 502 "real" (partly shorter) keywords appearing 1712 times. One third of the keywords was selected randomly, several of them differs only in the last phone (inflection). The *Radio* set contains 268 keywords randomly selected and appearing 1595 times. The *SpDat* set contains 606 keywords randomly selected and appearing 3821 times.

[2]Containing some signaling and DTMF tones. The calls have split channels (recorded in stereo, each conversation side was in separate channel).

[3]http://www.fee.vutbr.cz/SPEECHDAT-E/

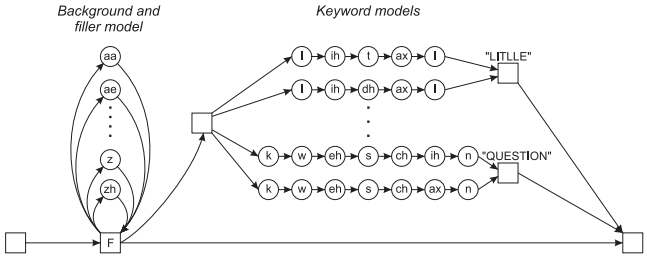

Fig. 7. Example of keyword spotting network.

Evaluation: Decision of hit / false alarm was done on phonetic alignment. So if a sequence of phones representing keyword detection appeared also in underlying phone alignment, it was classified as hit (regardless of orthographic transcription). This reflects our task – *acoustic* keyword spotting.

We used pooled Figure-of-Merit (FOM) and Equal-Error-Rate (EER) metrics for evaluation of the keyword spotter accuracy. FOM [4] is the average of correct detections per 1, 2, ... 10 false alarms per hour. We can approximately interpret it as the accuracy of KWS provided that there are 5 false alarms per hour. By *"pooled"*, we mean that all detections of all keywords are gathered to one set and scored independently on the keyword label. The EER represents operation point of keyword spotter, with the same number of missed keywords as false alarms.

2.6. Keyword score calibration

The calibration of keyword scores is crucial for presenting the results to the end-user, who pools all detections together and sorts them according to the score: the complaint we received with the initial (non-calibrated) versions of the system was that many false alarms are polluting-up the detections with high scores.

We proposed a keyword score calibration, which takes into account phones of the keyword. Each phone has attached a parameter (score penalty), regulating its contribution to the overall detection score. Beside these phone parameters, one more parameter is incorporated – the number of phones in the keyword. The calibration can be expressed as:

$$S_c(KW) = S_{LR}(KW) + a(0)|KW| + \sum_{i=1}^{|KW|} a(kw(KW,i)), \quad (1)$$

where KW denotes particular occurrence of the keyword, $S_c(KW)$ is the calibrated score, $S_{LR}(KW)$ is the raw score (likelihood ratio normalized by the length of detection), $\mathbf{a}$ is a vector of parameters (costs), $|KW|$ is number of phones which the keyword consists of and $kw(KW,i)$ is a function

returning index of ith phone of the keyword. The vector $\mathbf{a}$ has size equal to the number of phones plus one and must be estimated.

We need an objective function to be minimized (or maximized) to estimate the parameters $\mathbf{a}$. Such objective function can be for example FOM, but we found, that this did not match end-user requirements: maximizing FOM may lead to overall better accuracy, but small changes in calibrated scores of pooled keyword detections are not necessarily reflected. This confuses the calibration training algorithm and the process of maximization of objective function fails. That is why we decided to define different objective function, which reflects end-user requirements and does not confuse the calibration training.

The end-user pools all keywords together and sorts them according to the score. Then he/she goes through the list (from the best detection) and checks the correctness of the detection. The number of false alarms found in top several hundred detections is kind of ad-hoc accuracy of the keyword spotter. This inspired us to propose the following objective function: let us have pooled set of keywords detections DET sorted in descending order. Each detection det from DET has attached information if it is a hit $fa(det) = 0$ or a false alarm $fa(det) = 1$. The number of hits in DET is denoted as $hits$. The objective function is:

$$cost = \sum_{i=1}^{hits} fa(DET(i)) \frac{hits}{i}. \quad (2)$$

The function $DET(i)$ returns detection with ith best score from the pool of detection. The explanation of this objective function is the following: Let us have 100 hits in our calibration training data. A false alarm on the first position in DET, has a value of 100, false alarm on the second position has 50, and on the third position 33.3. If only these three false alarms are present, the $cost$ is $100 + 50 + 33.3 = 183.3$. We want to minimize the $cost$. This clearly pushes the false alarms away from the highest positions in DET.

We used iterative *Levenberg-Marquardt algorithm* (LMA) to minimize non-linear function $cost$ and find the values of $\mathbf{a}$. We run *LMA* in several iteration, where we randomly change the initial gradient descent step. This prevents us from getting stuck in a local minimum. The calibration parameters are usually estimated reasonably in a few tens of iterations. However, we found certain non-homogeneity of $\mathbf{a}$ parameters; found solutions sometimes differed in orders. That is why we decided to incorporate regularization to make the estimation more stable:

$$0 = \sum_i a(i) + r_0, \quad (3)$$

$$1 = \sum_i |a(i)| + r_1, \quad (4)$$

where r_0 and r_1 are residua that should be also minimized during the estimation of $\mathbf{a}$. We end-up with objective function $cost + r_0 + r_1$.

The calibration (see the following section for results) was trained on *DEV* data and 2382 randomly selected keywords longer than 2 phones.

3. RESULTS

We describe our final keyword spotting system in this section. We also report improvements (evaluated in terms of FOM and EER) for the most significant changes.

Our final system contains 2-step VAD at the beginning. Segments marked as speech are virtually clustered to 60–80 seconds long chunks. VTLN and mean normalization is applied on each chunk. Normalized segments are then processed by neural network, decoder, score calibration and decision. The 60–80 seconds long chunks were found to be sufficiently robust to possible switch of speakers or change in background noise (tuned on the *DEV* set). Shorter chunks lead to worse estimation of VTLN factor, however longer chunks lead to slower adaptation (worse accuracy during eventual speaker change). We used fast estimation of VTLN proposed in [3]. The approach uses MAP adaptation from UBM (single GMM with 32 diagonal Gaussians trained on all data) to derive specific models for each warping factor. Models are retrained using MMI (Maximum Mutual Information) criterion. Features for this model are 13 PLPs including c_0 with deltas and double deltas (without any normalization).

We found the following important improvements during the development of our system (see Table 1 for the impact on FOM and EER):

64–3800 band We started with filter-bank limited by frequencies 300–3600Hz (standard telephone band). However, the data contains lots of mobile phone recordings. We observed significant improvement by setting wider frequency band.

Omitting variance normalization We also started with mean and variance normalization applied on critical bands. We found that using mean normalization only is beneficial. This is probably due to errors in variance estimation in case of sub-optimum VAD performance.

CTS-RR data All training data was not available at the beginning of the project. Adding more out of domain data had no significant impact on the accuracy evaluated on target data (*DEV* and *TEST*). On the other hand, the robustness was significantly improved (this can be seen on the out of domain test data).

Adding VTLN We found VTLN normalization improving our system.

Calibration Keyword calibration trained to push-out false alarms from the top hundreds of detections having the best score improved also significantly the FOM and EER: deeper analysis is given in the following section.

Table 1. Absolute improvements of FOM and EER (deterioration with minus symbol) for several modifications of the system.

		DEV	TEST	Radio	SpDat
final KWS system	FOM	91.03	74.90	58.38	88.75
	EER	25.25	42.35	47.84	40.82
64–3800 band	FOM	-	2.99	-	-
	EER	-	1.52	-	-
Omitting VN	FOM	-	3.04	-	-
	EER	-	2.69	-	-
CTS-RR data	FOM	0.47	-1.38	14.08	14.71
	EER	1.00	0.06	11.66	13.66
Adding VTLN	FOM	0.37	1.87	0.62	0.42
	EER	0.86	1.46	1.19	0.65
Calibration	FOM	0.21	2.13	1.98	1.89
	EER	1.15	2.51	1.69	2.36

4. DISCUSSION AND CONCLUSIONS

We have presented several steps done to obtain a usable acoustic keyword spotting system. The novelty of our system is especially in the calibration. We found, that objective function based on FOM does not lead to successful gradient descent in used iterative *Levenberg-Marquardt algorithm*. That is why, we aimed to search for better, end-user-motivated, cost function.

We plot an example of false alarm occurrences in sorted detections in Figure 8. Each false alarm is marked by a "stem". Small red color stems are generated by the non-calibrated system. Larger green color stems belong to the calibrated one. Notice vanishing of false alarms from the top range of scores (left side). This is important, as the top of the list is usually the most important output for the end-user.

The impact of calibration on ROC curve is shown in Figure 9. We zoomed the very beginning of the curve. Red ROC curve denotes non-calibrated system, green curve denotes calibrated one. We can see the most significant improvement of the calibration at the very beginning of the ROC curve, that is the most important in the real scenarios: consistent improvement of several percents of detection rate is achieved. The $cost$ function evaluated on the *DEV* set at the beginning of estimation of the $\mathbf{a}$ parameters was 58×10^4 and was minimized to 11×10^4. On the *TEST* set, non-calibrated system achieved $cost = 1507$ versus $cost = 1403$ for the calibrated one.

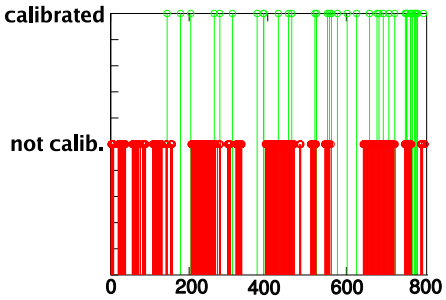

Fig. 8. False alarm occurrences in the first 800 detections in *DEV* set, sorted according to the score (x-axis). False alarms from non-calibrated system are in red, false alarms from the calibrated one in green.

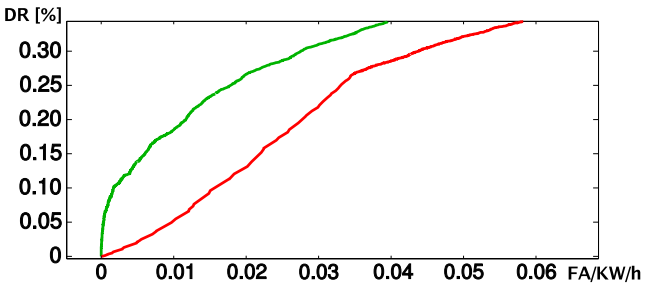

Fig. 9. Zoomed part of ROC curve on *DEV* set. ROC of non-calibrated system is in red, ROC of calibrated system is in green. X-axis is the number of false alarms per keyword per hour of data, Y-axis is the detection rate.

5. REFERENCES

[1] Szöke, I., Schwarz, P., Burget, L., Fapšo, M., Karafiát, M., Černocký, J. and Matějka, P., "Comparison of Keyword Spotting Approaches for Informal Continuous Speech", Interspeech, 633–636, Lisbon, 2005

[2] Grézl, F., Karafiát, M. and Burget, L., "Investigation Into Bottle-Neck Features for Meeting Speech Recognition", 10th Annual Conference of the International Speech Communication Association, 294–2950, Brighton, 2009

[3] Welling, L., Kanthak, S. and Ney, H., "Improved methods for vocal tract normalization", ICASSP, 764–764, Phoenix, 1999

[4] Rohlicek, J. R., Russell, W., Roukos, S. and Gish, H., "Continuous hidden Markov modeling for speaker-independent word spotting", ICASSP, 627–630, Glasgow, 1989