

Speaker recognition based on multilevel speech signal analysis on Polish corpus

Szymon Drgas · Adam Dabrowski

Published online: 25 May 2013

© The Author(s) 2013. This article is published with open access at Springerlink.com

Abstract This article deals with a new approach to the text-independent speaker verification task. It is namely proposed to combine spectral and the so-called high-level features (prosodic, articulatory, and lexical) in order to increase accuracy of speaker verification. The presented experiments were performed using a Polish language corpus developed by the authors, the so-called PUEPS corpus. It contains semi-spontaneous telephone conversations (acted emergency telephone notifications) recorded in laboratory conditions. As the Polish language is under resourced and the PUEPS corpus is relatively small, in this case a new approach is needed, other than these well known from NIST (National Institute of Standards and Technology) evaluations. The authors proposed to use the fast scoring instead of more complex classifiers and the AdaBoost (adaptive boosting) algorithm for features combination. Combination of features resulted in the equal error rate (EER) reduction for various SNR (signal-to-noise ratio) conditions. Additionally, score normalization methods were evaluated. It was shown that significant benefits can be obtained using the z-norm2 method.

Keywords Speaker recognition · High-level features · Kernel combination · Boosting

1 Introduction

In this article the text-independent speaker verification task is considered in the context of emergency telephone conversations. An approach that was chosen, combines

S. Drgas (✉) · A. Dabrowski
Division of Signal Processing and Electronic Systems, Chair of Control and Systems
Engineering, Poznan University of Technology, ul. Piotrowo 3A, Poznan, Poland
e-mail: szymon.drgas@put.poznan.pl

A. Dabrowski
e-mail: adam.dabrowski@put.poznan.pl

spectral and the so-called high-level features in order to increase the recognition accuracy [20]. The aim of this work is to show advantages of high-level features in the case of the Polish language. The high-level features carry information about the speaker especially in the case of the spontaneous speech. In order to perform this task database that contain Polish spontaneous speech was needed. Dabrowski et al. [9] developed Polish language corpus called PUEPS. It contains semi-spontaneous telephone conversations recorded in laboratory conditions. This gives a possibility to control degradation of the speech signal. As the Polish language is under resourced and the PUEPS corpus is relatively small, another approach is needed than these known from the well known NIST evaluations. In this paper a method is proposed that is a combination of algorithms known from the literature—cosine similarity system with scoring methods z-norm and z-norm2 (z-norm modified by authors), where features combination is performed by means of the AdaBoost algorithm. This method turned out to be effective for the corpus with Polish speech. The authors proposed to use the fast scoring instead of more complex classifiers and the AdaBoost algorithm for features combination. Moreover, the use of various scoring methods has been investigated.

The article is structured as follows: in Section 2 multi-level features approach to speaker recognition is described. This is followed by Section 3 in which SVM-based speaker recognition is reviewed. Next, in Section 4, the scoring and feature combination methods are presented. In Section 5 an experimental method, description of the features, and the results are shown and discussed. Finally, conclusions are given in Section 6.

2 Multi-level speaker recognition

In the multi-level speaker recognition systems several types of features are extracted, next they are combined, and finally a classifier is used to discriminate speakers. The so-called higher level features provide information complementary to classic spectral features and they make the system more robust [3, 6, 11, 18, 20]. In this work four types of features were used: spectral, prosodic, articulatory, and lexical. Spectral features convey information about timbre, which is related to shape and size of the vocal tract. They are based on computation of MFCC (mel-frequency cepstral coefficient) features. Prosodic features carry information about intonation, accent, and rhythm. These features are obtained from F0 and intensity contours. Articulatory features are related to the characteristic pronunciation of the speaker. In order to extract these features, neural networks trained to discriminate between articulatory classes were employed. Finally, lexical features correspond to characteristic words or phrases used by the speakers. In this work only pairs of consecutive words are considered (bigrams). They were obtained from transcriptions of the utterances in the PUEPS corpus.

3 Speaker recognition using support vector machines

Support vector machines (SVM) [22] are two-class hyperplane based classifiers that became successful in the text-independent speaker recognition [15]. These classifiers have good generalization property thanks to maximum-margin training criterion. In order to obtain nonlinear decision boundary, so called kernel trick can be applied.

The data from the input space can be nonlinearly transformed via mapping $\phi(\cdot)$ to the feature space. The mapping is performed implicitly, by specifying an inner product for each pair of points $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i)\phi(\mathbf{x}_j) \rangle$ rather than giving coordinates in the feature space. The information that is needed to train the SVM is kernel $N \times N$ matrix \mathbf{K} which contains all pairwise evaluations of kernel function $\kappa(\cdot, \cdot)$. The decision boundary can be written [21]:

$$f(\mathbf{x}) = \sum_{i=1}^N \alpha_i y_i \kappa(\mathbf{x}_i, \mathbf{x}) + b \quad (1)$$

where α_i are decision boundary coefficients (Lagrange's multipliers) and b is bias term. The α coefficients are determined during optimization where objective function is:

$$\begin{aligned} & \underset{\alpha_1, \dots, \alpha_N}{\text{maximize}} \quad \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \\ & \text{subject to} \quad 0 \leq \alpha_i \leq c \quad i = 1, \dots, N \end{aligned} \quad (2)$$

$$\sum_{i=1}^N \alpha_i y_i = 0,$$

where y_i is label of i 'th example. c is the regularization parameter for setting the tradeoff between maximum-margin and empirical error criteria.

One of the most successful acoustic features for speech recognition are mel-frequency cepstral coefficients. Each recording can be represented as a sequence of vectors. The typical rate is 100 vectors per second of the input signal. In order to be able to classify recordings using SVM each recording is transformed to the one fixed-length vector. One method to do this is GMM supervector [5]. Where for each recording GMM model is trained and then each conversation is represented by the vector that is concatenation of means of GMM model.

The kernel function to be valid has to fulfill Mercer's condition [19]. This means that function $\kappa(\cdot, \cdot)$ can be decomposed into inner product of its transformed parameters— $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i)\phi(\mathbf{x}_j) \rangle$. When a kernel function fulfills Mercer's condition then the corresponding kernel matrix is semi-positive definite.

In order to combine information from multiple sources, multiple kernel learning (MKL) algorithm can be applied [17]. The idea of this algorithm is to use kernel matrix in the following form

$$\mathbf{K} = \sum_{i=1}^K \beta_i \mathbf{K}_i \quad (3)$$

and optimize coefficients β_i and α_i concurrently. The optimization is performed alternately for a given β_i parameters α_i coefficients are optimized using standard SVM learning algorithm, and next β_i coefficients are optimized using gradient method. This is continued alternately until convergence.

4 Fast scoring and feature combination

In the present study the corpus that consists of telephone conversations has been used. These conversations mimic the emergency telephone notifications. From each conversation segments of signal that contain the speech of the calling person were extracted. The set of segments extracted from the signal of a given speaker from a given conversation is called conversation side.

In the presented speaker recognition system after the feature extraction phase, each conversation side is represented as K vectors. As in this work spectral, prosodic, articulatory, and lexical features were used $K = 4$. These vectors are stored in K matrices $\mathbf{X}_1, \dots, \mathbf{X}_K$ where:

$$\mathbf{X}_k = [\mathbf{x}_{k1} \ \dots \ \mathbf{x}_{kN}], \quad (4)$$

where \mathbf{x}_{kj} is the feature vector of j 'th type extracted for j 'th conversation side. N denotes the number of the conversation sides available in the corpus. In the next phase, kernel matrices were computed from matrices \mathbf{X}_k . In the performed experiments cosine kernel was used. The cosine kernel is a function that for a given pair vectors returns an inner product of these vectors normalized to the unit length. These inner products for all pairs of available vectors are collected in the kernel matrix. The (i, j) element of the matrix \mathbf{K}_i can be calculated as follows:

$$(\mathbf{K}_k)_{ij} = \frac{\mathbf{x}_{ki}^T \mathbf{x}_{kj}}{\|\mathbf{x}_{ki}\| \|\mathbf{x}_{kj}\|}. \quad (5)$$

It can be treated as a measure of similarity between i 'th and j 'th conversation sides for the features of type k .

There is no SVM classifier used, but values of the kernel matrix elements are used as decision scores. This can cause a problem when there are more than one training side per speaker model. It is not obvious how to combine information from the available training sides in order to achieve higher speaker recognition accuracy. In this article, this problem has not been addressed. It is because durations of conversation sides collected in the PUEPS corpus are not normalized. This is caused by a type of conversations. Indeed, it is natural that the emergency telephone calls differ in duration. The experiments with varying number of training sides could lead to uninterpretable results, when the times of these sides differ. Another argument for this approach is the following one: when there is no large dataset available to model background speakers, there is no benefit from using SVM in terms of the verification error. Thus, its rejection simplifies the system and reduces the computational complexity. Using kernel matrix element values directly as scores is known in the literature as the fast scoring [10].

In case of the fast scoring method only information from the two compared conversation sides is taken into account. In order to include some global information, the score normalization can be performed [2]. These methods reduce the differences in scores for different models. In the literature there exist well known methods such as z-norm and t-norm. In this work z-norm has been considered. It can be said that fast scoring without score normalization is a speaker vs speaker verification system. When scores are normalized it is speaker vs all—the information based on the modeled speaker and all other available speakers is stored in each model.

An idea of the z-norm method is to test the model i using a set of impostor sides B . Next, the statistics (mean $\mu_z^{(i)}$ and variance $\sigma_z^{(i)}$) are computed from the resulting set of scores $\{(\mathbf{K}_k)_{ij}\}_{j \in B}$. Next, the scores are shifted and scaled using these obtained statistics

$$(\bar{\mathbf{K}}_k)_{ij} = \frac{(\mathbf{K}_k)_{ij} - \mu_z^{(i)}}{\sigma_z^{(i)}} . \tag{6}$$

It is important to use a proper kernel in such systems. Each kernel matrix element should reflect similarity between the corresponding sides. For example, it is possible to show datasets, for which linear kernel (i.e. inner product of vectors without normalization) values would not reflect similarity. Instead, the cosine kernel or spherical normalization can be used.

Using fast scoring makes it possible to use the boosting algorithms to combine information from different features. However, values of elements of the kernel matrix should be centered at zero. Thus, the score matrix \mathbf{S} has been introduced

$$\mathbf{S}_k = \bar{\mathbf{K}}_k - \theta_k^{\text{EER}} \mathbf{1}\mathbf{1}^T . \tag{7}$$

where $\mathbf{1}$ is a vector filled with ones, while θ_k^{EER} is the threshold, for which probability of false alarms is equal to the probability of the miss error for k 'th kernel matrix. Each kernel matrix needs to be shifted by this value. After this operation the semi-definite property of the kernel matrices can be lost.

The t-norm method works in a similar way, with a difference that every test side is tested with a set of impostor models and then shifted and scaled. In preliminary experiments this method led to a relatively high error increase. Thus, it was not used in further tests.

After the score normalization when the model side is compared with itself, different results can be obtained. In the reported experiments an additional variant has been evaluated. The scores for a given model were normalized in such a way that $(\hat{\mathbf{K}}_k)_{ii} = 1$. This case is denoted as the z-norm2.

Kernel boosting is a method of the kernel matrix combination, which is similar to the MKL algorithm and the combination coefficients are optimized

$$\mathbf{S} = \sum_{k=1}^K \lambda_k \mathbf{S}_k , \tag{8}$$

where λ_k coefficients determine contributions of features in the resulting kernel matrix. Boosting is based on the AdaBoost algorithm [8]. In the present work score matrices \mathbf{S}_k instead of the kernel matrices have been used.

The algorithm can be divided into the following steps:

1. **Input:** Score matrices set and the corresponding labels: $\{(\mathbf{S}_k, \mathbf{D})\}_{k=1}^K$, where

$$(\mathbf{D})_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_{ki} \text{ and } \mathbf{x}_{kj} \text{ belong to the same class} \\ -1 & \text{otherwise} \end{cases}$$

2. **Initialization:** $\mathbf{W} = 1/m$, where $m = \frac{N(N+1)}{2}$, $\hat{\mathbf{S}} = 0$.

3. For each pair $(\mathbf{S}_k, \mathbf{D})$

- (a) $S^+ = \{(i, j) : (\mathbf{S}_k)_{ij}(\mathbf{D})_{ij} > 0\}$, $S^- = \{(i, j) : (\mathbf{S}_k)_{ij}(\mathbf{D})_{ij} < 0\}$
- (b) $W^+ = \sum_{(i,j) \in S^+} (\mathbf{W})_{ij} |(\mathbf{S}_k)_{ij}|$
 $W^- = \sum_{(i,j) \in S^-} (\mathbf{W})_{ij} |(\mathbf{S}_k)_{ij}|$
- (c) $\lambda_k = \frac{1}{2} \log \left(\frac{W^+}{W^-} \right)$
- (d) $(\mathbf{W})_{ij} = (\mathbf{W})_{ij} \exp(-\lambda_k (\mathbf{D})_{ij} (\mathbf{S}_k)_{ij})$
- (e) $\mathbf{W} = \frac{\mathbf{W}}{\mathbf{1}^T \mathbf{W} \mathbf{1}}$ where $\mathbf{1}$ is a column vector filled with ones, $\mathbf{1}^T \mathbf{W} \mathbf{1}$ is a sum of all elements of the matrix \mathbf{W} .
- (f) $\hat{\mathbf{S}} = \hat{\mathbf{S}} + \lambda_k \mathbf{S}_k$.

4. Output: Kernel matrix $\hat{\mathbf{S}}$.

Boosting algorithm uses exponential loss function $\exp(-x)$ that bounds empirical error. This bound B can be written as

$$B = \sum_{i,j} \exp[-(\mathbf{S})_{ij}(\mathbf{D})_{ij}]. \quad (9)$$

The matrix \mathbf{W} contains the weights for the trials that are used for optimization. For the first type of features it is set uniformly e.g. all trials have the same weight. The number of distinct trials (the order of recordings in a trial is irrelevant) is $\frac{N(N+1)}{2}$. In each iteration two sets that contain index pairs that represent trials are determined: S^+ and S^- . S^+ contains pairs correctly classified using current score matrix $\hat{\mathbf{S}}$ while S^- contains badly classified trials. In the next step, two scalar values are computed: W^+ and W^- , they are the sums of weighted scores of correctly and badly classified trials respectively. This is followed by calculating of λ_k weight of the current score matrix. This calculation is optimal in a sense of the error bound B [8]. In the next phase weights in the matrix \mathbf{W} are updated. The higher weights are assigned to the trials with the worst scores. It is followed by the normalization step in order to guarantee that elements of matrix \mathbf{W} sum up to 1. This process is repeated for each matrix \mathbf{S}_i . Using this function the optimal kernel weights are determined. Additionally, each element of the added kernel matrix is weighted in order to optimize mixing weight mainly using badly classified data.

5 Experiments

5.1 PUEPS corpus

PUEPS corpus was recorded in order to provide spontaneous speech samples in Polish. It contains a set of telephone conversations with the emergency telephone service.

5.1.1 Data recording and system architecture

There were several requirements while PUEPS corpus [4] was designed. The language of speech had to be Polish. Second, there was a need for spontaneous speech. It has been very important as lexical features have sense when the speaker uses

his/her own words. It has also significance in case of prosodic features (tempo and speaking style are less consciously controlled by a speaker) as well as articulatory (speaker articulates words less carefully). Another requirement of this corpus was a laboratory quality of the recordings. This feature allows free signal degradation; for example using Head and Torso simulator as in [7]. Finally, each speaker needed to record several conversations to make automatic speaker recognition experiments possible. Moreover, the time between conversations of one speaker should be longer than a week in order to obtain a within-speaker variability that is close to reality.

The records contain the acted telephone conversations with the emergency telephone services. The task of the speakers was to act as a person that is a victim or observer of a crime or an emergency situation. He/she had to report the event to the emergency telephone operator. Only Polish native speakers took part in the recordings. In order to provide the speaker information about the situation to describe, without suggesting words to use, video movies were prepared and presented to the speakers.

At the beginning of the experiment the speaker had to watch such a short movie with a crime scene. After that he or she was calling to the emergency phone and reporting a crime to a person who played role of the police officer. The telephone call was recorded and prepared for the post-processing.

The caller was located in an anechoic room equipped with a terminal, a telephone, and a high quality microphone. After watching the movie the participant called over the PBX (private branch exchange) to the emergency phone operator, who was located in another laboratory. The call was recorded twofold: with a digital call recorder from the telephone line and with an audio recorder from the observer microphone. The call recorder processed signal of the whole conversation with a typical telephone line quality, while the high quality caller voice was stored with the audio recorder for post-processing.

5.1.2 Calls database statistics

30 speakers participated in the recording sessions of the PUEPS corpus. All of them were students aged between 19 and 26. Majority of recordings are recordings of male voices (27 speakers) with the rest done by females (3 speakers). The role of the operator that received emergency notifications was performed by 3 persons.

Each speaker recorded 6 conversations. Two conversations maximum were recorded by one speaker during one session. Minimal time between two sessions in which a speaker could take part was one week. The mean conversation time is 111 s.

5.2 Feature extraction

The high-level features were extracted in the same way as in the earlier authors' work [12]. The shortened description of extraction is presented below.

5.2.1 Spectral features

In order to convey spectral information to multilevel recognition system, GMM (Gaussian mixture models) supervectors were used. First, the MFCC features were extracted. The frame length was set to 20 ms and frame step to 10 ms. The number

of mel-frequency filters was 30 and first 20 cepstral coefficients were saved for further processing. Next, for each recording the GMM model was trained. The number of components was 30. Each component was represented by the Gaussian function with the diagonal covariance matrix.

In order to cope with two distributions the Kullback–Leibler (KL) divergence was used. However, it cannot be applied directly, because in our case the Mercer's condition would not be fulfilled. One of possibilities is to use the function that is an upper bound of the KL divergence

$$d(\mathbf{m}_1, \mathbf{m}_2) = \frac{1}{2} \sum_{i=1}^C \omega_i (\mathbf{m}_i^1 - \mathbf{m}_i^2)^T \Sigma_i^{-1} (\mathbf{m}_i^1 - \mathbf{m}_i^2), \quad (10)$$

where ω_i denotes weights, \mathbf{m}_i mean, and Σ_i is a covariance matrix of the i 'th component. C is the number of GMM components. Weights and covariance matrices are the same for all conversations and are equal to parameters of the global GMM model for the training data. Kernel function can be expressed in the following way:

$$K(\mathbf{m}_1, \mathbf{m}_2) = \sum_{i=1}^C \omega_i \mathbf{m}_i^1 \Sigma_i^{-1} \mathbf{m}_i^2. \quad (11)$$

The vector that represents i 'th conversation side has the following structure

$$\mathbf{v}_i = \begin{bmatrix} \sqrt{\omega_1} \Sigma_1^{-2} \mathbf{m}_1 \\ \sqrt{\omega_2} \Sigma_2^{-2} \mathbf{m}_2 \\ \vdots \\ \sqrt{\omega_C} \Sigma_C^{-2} \mathbf{m}_c \end{bmatrix} \quad (12)$$

As Σ_i matrices are diagonal, the matrix Σ_i^{-2} is also diagonal and contains the inverses of the square roots of elements. As 20 cepstral coefficients have been computed for each frame, and GMM has 30 components, the total number of elements of vector \mathbf{v}_i is 600.

5.2.2 Prosodic features

The prosodic features are based on a linear approximation of F0 and intensity contours computed similarly as in [1]. The procedure consists of the following steps:

1. fundamental frequency and intensity contours extraction,
2. determination of voiced speech segments,
3. approximation of F0 contours with lines,
4. approximation of intensity contours with lines,
5. quantization of directional coefficients and interval lengths,
6. combination of quantized sequences into one sequence,
7. calculation of a bag of n -grams statistics and forming the resulting vector.

The number of quantization levels of F0 slope with addition of unvoiced segment, intensity slope and segment duration was 2. Thus each segment could be coded

using one of 12 codes. However, in order to reduce the number of possibilities, for unvoiced segments there is no distinction between rising and falling intensity. Finally each segment was coded with one of 10 codes. Thus, in case of unigrams, the vector representing prosodic aspects of a given conversation side had 10 elements, while for bigrams it was 100.

5.2.3 Articulatory features

In order to catch speaker characteristics connected to pronunciation, articulatory features were extracted. It has been done using neural networks trained on 2000 h of the telephone speech [14]. First, spectral features were extracted using PLP (perceptual linear prediction) method. The PLP feature extraction was done in the following way using HTK software (hidden Markov toolkit) [23]. First preemphasis filter with coefficient equal to 0.97 was applied. Next, signal were divided into frames with 25 ms window length and 10 ms frame step. The number of frequency channels was 24 and linear prediction order was 12. Finally, linear prediction coefficients were transformed to the 12 cepstral coefficients. These 12 features together with log-energy coefficient formed basic feature vectors. These vectors were expanded by delta and delta-delta features. Finally feature vector consisted of 39 elements. Next, each frame and its context (4 preceding and 4 subsequent frames) has been classified using multi-layer perceptrons (MLPs) in order to extract articulatory features. Two categories of articulatory features were taken into account: place of articulation and articulatory degree (two MLPs were used). The perceptrons had 3 layers. The number of units of the input layer was 351 (dimension of the PLP feature vector times 9 - frame and its context), the number of hidden units was set according to the amount of the available data. It was 1900 for MLP for place of articulation and 1600 for degree of articulation. The outputs of the MLPs correspond to places and degrees of articulation summarized in Table 1. Degree of articulation network had 6 output and place of articulation network had 10 output. For each frame the output with the highest activation has been coded. Thus, degree of articulation had 6 symbols and place of articulation had 10 symbols. These sequences of symbols were combined into one sequence with $6 \times 10 = 60$ symbols. Finally statistics of bigrams have been computed. There were $60 \times 60 = 3600$ of possible bigrams.

5.2.4 Lexical features

Idiolectal aspects of speakers have been also taken into account [11]. They were caught by lexical features obtained from manually made transcriptions of the PUEPS corpus. First, dictionaries were constructed for word bigrams for the available data. Only the bigrams that occurred at least 4 times were used in the dictionary. This resulted with the dictionary with 425 entries. Then for each side the number

Table 1 Outputs of the MLPs for articulatory features extraction

Category	Outputs
Place	Alveolar, dental, labial, labio-dental, lateral, post-alveolar, rhotic, velar
Degree	Approximant, closure, flap, fricative, vowel

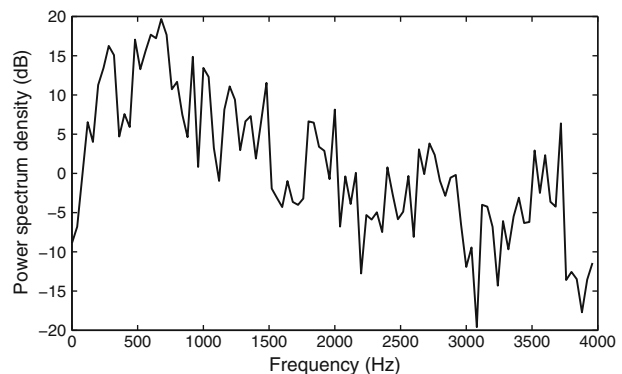
of occurrences of each bigram in the dictionary was counted. These numbers of occurrences were used to construct bigram vectors for each side, which resulted with 425-dimensional vectors.

5.3 Experimental setup

A fact that the PUEPS corpus was recorded in laboratory conditions gives a possibility to perform experiments, in which a degree of the signal degradation is controlled. The so-called “babble noise” was added to original recordings with the following SNRs: 20, 10, and 0 dB. The noise was added before limiting to the phone line bandwidth. The used babble noise was obtained by summing many sentences in Polish languages [16]. These sentences were shifted and some of them were reversed prior to mixing. All mixed sentences had normalized level. The power spectrum density of the noise is presented in Fig. 1. It reveals maximum between 500 and 1000 Hz. It has also a second maximum between 3500 and 4000 Hz which reflects a consonants as Polish is a sort of “consonant” language [16].

The SNRs were calculated using active speech level obtained with ITU.T P56 norm. For testing the verification error for each feature type individually the data from all speakers from the PUEPS corpus were used. However, in the case of testing combinations of features it was necessary do divide the corpus into the background and the test data (15 speakers for the background and 15 speakers for the test). The background data were used to train feature weights (using AdaBoost). The system has been evaluated in terms of equal error rate (EER). During the test system had to decide whether a given pair of recordings (trial) contain the speech of the same speaker. There are two kinds of trials: true trials (where speakers in recordings match) and false trials. The number of badly classified true trials is miss error, while badly classified false trials result in false acceptance error. These errors depends on the threshold. The error for which these two types of errors are equal is called EER. As the number of speakers in the corpus is relatively small, the experiments were performed for 20 different divisions of speakers. This was motivated by the fact that, are relatively short for high-level features. Additionally there is some variance in the lengths of recordings. Such results multiplication gives an opportunity to check how data division influences the results. The divisions were selected randomly. EERs from all divisions were averaged.

Fig. 1 Power spectrum density of babble noise



5.4 Results

The DET (detection error tradeoff) curves for SNR 20 dB are presented in Fig. 2. The lowest EER (9.26 %) was achieved for spectral features. The about two times higher EER has been brought for prosodic features (21.72 %). A slightly worse accuracy was obtained for lexical features (29.14 %). The worst performance has been observed for articulatory features (EER = 33.50 %).

For spectral features the EER is about two percent higher than in case of experiments conducted by the authors with the Switchboard corpus [13] for one training side condition. The cause of this discrepancy may be the following factors: first, duration of each conversation side and second, presence of the background dataset. For the Switchboard database the side duration it is about 2.5 min while for the PUEPS corpus the average duration is 1 min and its dispersion is much higher than this of the Switchboard database. Additionally, for the PUEPS corpus no background dataset is available.

Prosodic features, despite the duration and time differences, give similar accuracies. The prosodic features vector has a dimension of 100. This is not a very large number and it can perform well.

Articulatory features for the PUEPS corpus give a higher error than that in case of the Switchboard database [13] and it is equal to 33.5 %. Here, the number of elements of the feature vectors is 3600. In case of short conversations (about 1 min) this number is similar to the number of frames for many conversations in the PUEPS corpus. This can lead to a high noise level.

Lexical features give error about 50 % higher than in case of the Switchboard. Here the bigram lexicon contains 424 positions.

The DET curves for the case, in which SNR is 10 dB, are presented in Fig. 3. The increase of the noise level only in a small degree influenced the EER for spectral

Fig. 2 Results for SNR 20 dB

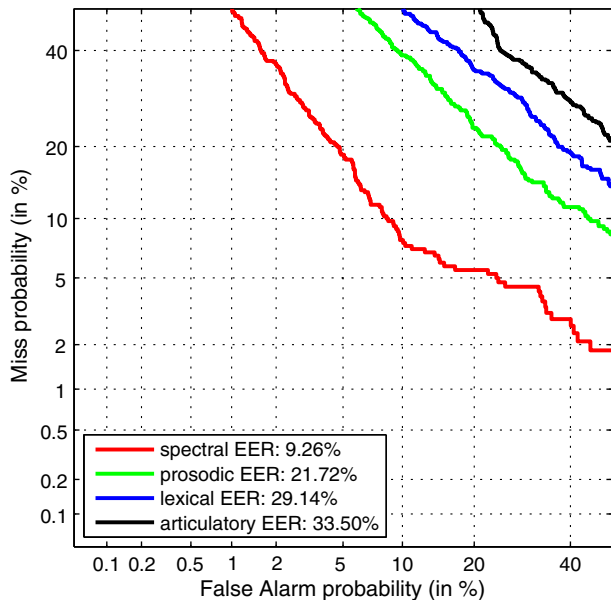


Fig. 3 Results for SNR 10 dB

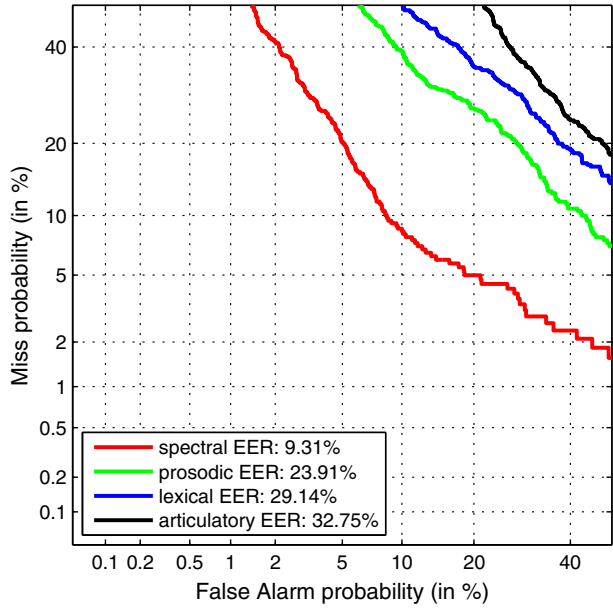


Fig. 4 Results for SNR 0 dB

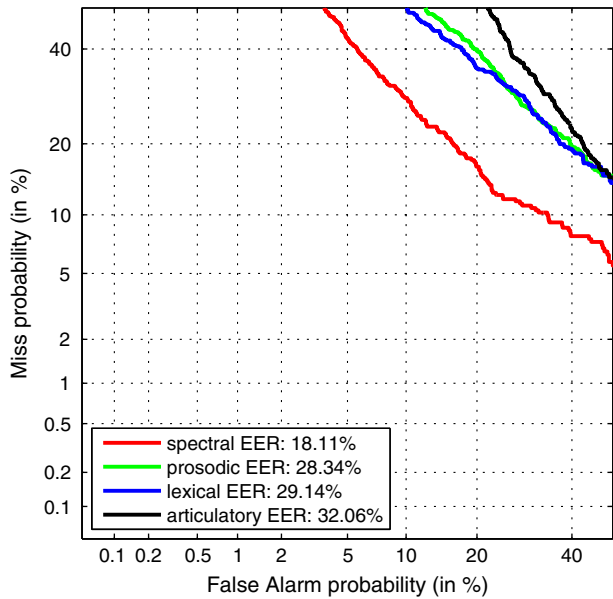


Table 2 EERs (%) obtained for spectral features for various training and testing conditions

Training SNR	Testing SNR		
	20 dB	10 dB	0 dB
20 dB	9.32	13.01	34.24
10 dB	14.31	9.40	27.75
0 dB	39.35	30.57	18.58

Table 3 EERs (%) obtained for prosodic features for various training and testing conditions

Training SNR	Testing SNR		
	20 dB	10 dB	0 dB
20 dB	24.45	31.27	43.99
10 dB	28.37	25.89	39.96
0 dB	44.01	41.70	32.47

features. It increased by about 0.05 %. Prosodic features turned out to be more sensitive to “babble noise”—it increased by about 10.08 %. For articulatory features with SNR decrement the EER slightly decreased. It can be caused by masking some unreliable features.

For SNR 0 dB (see Fig. 4) significant deterioration of accuracy of spectral features was observed—the EER increased 2 times in comparison to the SNR = 10 dB case. For prosodic features the EER increased by about 18,53 %. The EER for articulatory features remained at a similar value, (above 30 %), but it is still high. This means that it does not carry much additional speaker-dependent information.

There were also performed experiments for mismatched SNRs in the training and the test sides. The results obtained for spectral features are shown in Table 2. It can be noticed that errors dramatically increased in the case of the SNR mismatch—the higher difference in SNRs the higher the error. Another observation is that the lower error is obtained when the model side has higher SNR than the test side.

The results for mismatched noise degradation for prosodic features are shown in Table 3. Similarly to the spectral features case the error increases, when noise levels are different in the model and the test samples. However, the relative difference is lower than in the previous case. Thus it can be said that the prosodic features are less sensitive to additive noise than the spectral features.

5.4.1 Combined features

The results of the speaker verification accuracy based on features combination with weights of each feature set, determined with the AdaBoost algorithm, are presented in Table 4. In the experiments the spectral, prosodic, and lexical features were combined. The articulatory features were rejected, because of a relatively high EER (more than 30 %). Indeed, the preliminary experiments showed that the considered combination together with articulatory features led to some system accuracy deterioration.

For SNR = 20 % the system based on features combination gives an error of 21.21 % lower than for the system based on spectral features. For SNR = 10 dB an improvement caused by the features combination is slight—it is about 5 %. It is caused by the fact, that the noise influenced spectral features in small degree while

Table 4 EERs (%) obtained for various score normalization techniques

SNR	Without normalization		z-norm		z-norm2		SVM baseline
	Spectral	Combined	Spectral	Combined	Spectral	Combined	Spectral
20	9.59	7.54	10.12	7.17	6.57	4.73	7.67
10	9.92	9.38	10.76	8.54	6.37	6.06	8.22
0	18.89	16.39	16.60	14.34	15.28	13.40	14.78

Table 5 EERs (%) obtained for combined features for various training and testing conditions

Training SNR	Testing SNR		
	20 dB	10 dB	0 dB
20 dB	4.73	6.42	21.63
10 dB	8.83	6.06	19.75
0 dB	28.90	22.21	13.40

accuracy of the prosodic features was decreased. In this case, a combination with high-level features caused the EER decrement by 13.23 %.

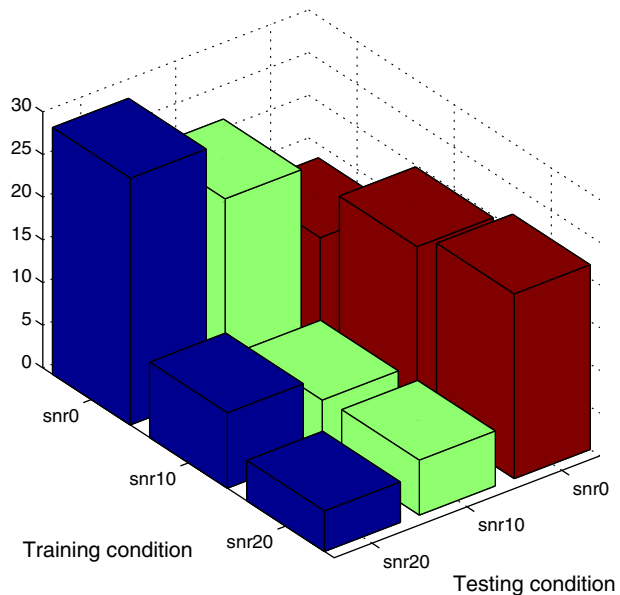
When z-norm was applied before combination of the features the error increased for spectral features tested individually for high SNRs. However, for combined features considerable benefit was obtained for the whole range of noise degradation.

The best results were obtained in case of z-norm2 normalization. It gave higher relative EER reductions in all tested cases (individual and combined).

The last column of Table 4 contains the results obtained for the SVM baseline system with GMM supervector kernel. It can be noticed that z-norm2 method outperforms the baseline for spectral features for SNRs 20 and 10 dB. In case of SNR 0 dB the SVM results are slightly better than z-norm2 for the spectral features. However, when high-level features are applied the results are better than for the baseline.

As it was mentioned in Section 5.3 the dataset had to be divided into development and evaluation parts. There was 20 different divisions in order to check the variance of error reduction. In case of SNR 20 dB the mean error reduction (absolute) was 1.65 %. In all divisions there was an EER reduction standard deviation of this error reduction was 0.8 %. In case of SNR 10 dB, although the mean error reduction was 0.8 % there were some divisions in which combination with high-level features have not resulted in improvement. It can be the result of the fact that when SNR is lowered

Fig. 5 Combined features for mixed conditions



from 20 dB to 10 dB the prosodic features are more affected than spectral features. In case of SNR 0 dB the mean error reduction was 1.53 % while standard deviation of this error is 1.08 %. In all divisions combination with high level features resulted with EER reduction. It can be the consequence of the fact that lowering SNR from 10 dB to 0 dB, affects in higher degree spectral features than high-level features.

Finally, the results obtained with combined features in mismatched SNR conditions are presented in Table 5 and in Fig. 5. It turned out that the signal degradation mismatch leads to an error increase and better results are obtained when the model is trained using samples with better quality.

6 Conclusions

From the performed experiments the following conclusions can be drawn:

1. prosodic and lexical features provide a complementary information to spectral features in the speaker verification task
2. articulatory features extracted as present in the article do not provide sufficient amount of speaker-dependent information in multi-level speaker recognition systems. It can be a consequence of too high number of parameters in relation to lengths of the conversations. Further research concerning constraining of the articulatory feature extraction process are planned by the authors
3. AdaBoost and fast scoring methods can effectively be used to combine information from various features
4. additional score normalization (z-normnorm method) gives considerable error reduction
5. for mismatched degradation of training and test sides, significant error increase was observed.
6. the error is higher when the model side has lower SNR than the test side.

Open Access This article is distributed under the terms of the Creative Commons Attribution License which permits any use, distribution, and reproduction in any medium, provided the original author(s) and the source are credited.

References

1. Adami AG (2007) Modelling prosodic differences for speaker recognition. *Speech Commun* 49(4):277–291
2. Auckenthaler R, Carey M, Lloyd-Thomas H (2000) Score normalization for text-independent speaker verification systems. *Digit Signal Process* 10:42–54
3. Baker BJ (2008) Speaker verification incorporating high-level linguistic features. PhD thesis, Queensland University of Technology
4. Balcerek J, Drgas S, Dabrowski A, Konieczka A (2009) Prototype multimedia database system for registration of emergency situations. In: SPA conference
5. Campbell WM, Sturim DE, Reynolds DA (2006) Support vector machines using gmm supervectors for speaker verification. *Signal Process Lett* 13(5):308–311
6. Campbell W, Campbell J, Gleason T, Reynolds D, Shen W (2007) Speaker verification using support vector machines and high-level features. *IEEE Trans Audio Speech Lang Process* 15(7):2085–2094
7. Cetnarowicz D, Drgas S, Dabrowski A (2010) Speaker recognition system and experiments with head/torso simulator and telephone transmission. In: Signal processing algorithms, architectures, arrangements, and applications conference proceedings (SPA), pp 99–103

8. Crammer K, Keshet J, Singer Y (2002) Kernel design using boosting. In: NIPS 2002
9. Dabrowski A, Drgas S, Pawlowski P, Balcerek J (2012) Development of pueps—corpus of emergency telephone conversations. In: Proceedings of language resources for public security applications workshop, LREC 2012
10. Dehak N, Dehak R, Kenny P, Brümmer N, Ouellet P, Dumouchel P (2009) Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification. In: Interspeech 2009
11. Doddington G (2001) Speaker recognition based on idiolectal differences between speakers. In: Eurospeech-2001, pp 2521–2524
12. Drgas S, Dabrowski A (2011) Kernel alignment maximization for speaker recognition based on high-level features. In: Interspeech 2011, pp 489–492
13. Drgas S, Dabrowski A (2011) Kernel matrix size reduction methods for speaker verification. In: 5th language & technology conference
14. Frankel J, Magimai-Doss M, King S, Livescu K, Cetin O (2007) Articulatory feature classifiers trained on 2000 hours of telephone speech. In: Interspeech 2007
15. Kinnunen T, Li H (2010) An overview of text-independent speaker recognition: from features to supervectors. *Speech Commun* 52(1):12–40. doi:[10.1016/j.specom.2009.08.009](https://doi.org/10.1016/j.specom.2009.08.009)
16. Ozimek E, Kutzner D, Sek A, Wicher A (2009) Polish sentence tests for measuring the intelligibility of speech in interfering noise. *Int J Audiol* 48:433–443
17. Rakotomamonjy A, Bach F, Canu S, Grandvalet Y (2008) Simplemkl. *J Mach Learn Res* 9:2491–2521
18. Reynolds D, Andrews W, Campbell J, Navratil J, Peskin B, Adami A, Jin Q, Klusacek D, Abramson J, Mihaescu R, Godfrey J, Jones D, Xiang B (2003) The supersid project: exploiting high-level information for high-accuracy speaker recognition. In: Proc. IEEE international conference on acoustics, speech, and signal processing (ICASSP '03), vol 4, pp IV–784–IV–787
19. Shawe-Taylor J, Christianini N (2004) Kernel methods for pattern analysis. Cambridge University Press, Cambridge, UK
20. Shriberg E (2007) Higher-level features in speaker recognition, in speaker classification I. In: Lecture notes in computer science/artificial intelligence. Springer, Heidelberg/Berlin
21. Vapnik V (1995) The nature of statistical learning theory. Springer
22. Vapnik V (1998) Statistical learning theory. Wiley, New York
23. Young SJ, Evermann G, Gales MJF, Hain T, Kershaw D, Moore G, Odell J, Ollason D, Povey D, Valtchev V, Woodland PC (2006) The HTK book, version 3.4. Cambridge University Engineering Department, Cambridge, UK



Szymon Drgas is an assistant at the Department of Computing, Poznan University of Technology. He received M.Sc. degree in computer science from the Physics Department, Adam Mickiewicz University, Poznan and M.Sc. degree in physics also from Adam Mickiewicz University. In 2013 he received Ph.D. degree in automation and robotics from Poznan University of Technology. He is interested in speech processing and psychoacoustics.



Adam Dabrowski is a full professor in digital signal processing at the Department of Computing and head of the Division of Signal Processing and Electronic Systems, Poznan University of Technology, Poland. His scientific interests concentrate on: digital signal and image processing (filtering, signal separation, multirate and multidimensional systems, wavelet transformation), multimedia, biometrics, visual systems, and processor architectures. He is author or co-author of 4 books and over 300 scientific papers.

He was a Humboldt Foundation fellow at the Ruhr-University Bochum (Germany), visiting professor at the ETH Zurich (Switzerland), Catholic University in Leuven (Belgium), University of Kaiserslautern (Germany), and the Technical University of Berlin (Germany).

Currently he is Chairman of the Signal Processing (SP) and Circuits & Systems (CAS) Chapters of the Poland IEEE Section.

In 1995 Professor Adam Dabrowski won the IEEE Chapter of the Year Award, New York, USA. In 2001 he was also awarded with the diploma for the outstanding position in the IEEE Chapter of the Year Contest.