Speaker Diarization Based on Speech Signal Approximation by Step Function

Rustam Latypov, Evgeni Stolov
Kazan Federal University
Kazan, Russia
Roustam.Latypov@kpfu.ru

***Abstract*—In this paper, we describe a new method for speaker clustering in an audio file. The main idea is to replace the speech signal with a step function having a limited number of levels. The research goal is to determine the signal characteristics obtained from the analysis of the step function produced. The step function is created by setting multiple levels that divide the signal range into non-overlapping strips. All the source signal values inside a strip are changed for the strip's mark. Using the sine function as a template, we get recommendations for choosing the source's best-keeping features. We employ the obtained results to solve the problem of speaker diarization. The developed diarization algorithm requires little computer resources. The experiments show the suitability of the developed results to the conditions of the online diarization process.**

I. Introduction

Speaker diarization algorithms are used for various tasks, such as recording live news, recording meetings, and other applications. When solving this problem, speech segments corresponding to a particular speaker are assumed to have shared features unique to that speaker. Thus, the problem usually leads to finding a suitable signal representation and the right metric. Speaker diarization can be viewed as a particular case of speaker identification. The former's peculiarities are the possibility of working in online mode, a small number of speakers under consideration, and insignificant resources required for functioning.

As new speech recognition systems develop, it becomes possible to make the generated text more readable. Using speaker diarization, one can make each fragment of text accompanied by a mark of the speaker. Another application is the broadcast of the discussion. The camera should be facing the active speaker. In the case of broadcasting discussions online, the algorithm should require as few resources as possible.

Our methods presented in this paper are easy to implement since they use simple calculations, and some of them can be done in advance. In this paper, we consider the case when the fragments spoken by different speakers do not overlap.

Each system dedicated to solving a specified problem should contain the following main parts [1].

- Speech detection. We have to select only those fragments from the file, which contain a speech. Usually, this problem is solved by various voice activity detectors (VAD).
- Clustering; One has to collect all fragments belonging to one author. First of all, one has to extract the unique properties of the speaker. Many modifications of the Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM) are used to this end. However, recently, methods based on neural net are employed. In [2], the features are presented by d-vectors that are extracted by applying long short term memory (LSTM) to voiced fragments. The same scheme is explored in [3], but the most attention is dedicated to automatically detecting the number of speakers in a file. The other part of the clustering procedure calculates a distance between two features obtained in the previous step. In [4], a procedure for distance calculation based on the neural net is proposed. In [5], video information is added to the features extracted by standard methods.

In [6], leverage of form of the wave for speech separation is proposed. In this paper, we also implement an approach based on the investigation of the waveform. The main idea of our paper is as follows. We select two levels and intersect speech wave with two horizontal lines (see Fig. 1).

Using the sine function as a template, we propose inferencing local features of the signal based on lengths of the intervals *Len1* and *Len2* and their reciprocal arrangement. Another simple feature of wave used in the diarization procedure is the instant frequency calculated at maximal or minimal wave points. The standard algorithm for evaluating instant frequency based on the Hilbert transform's discrete version requires significant resources [8].

Since the analytic representation of a signal can construct the signal's envelope, any parameter related to the analytic form carries a signal feature. The authors proposed a very light algorithm to evaluate the instant frequency at extreme points [12]. That frequency is used for the creation hash, leveraged for distinction speech fragments belonging to different speakers.

This paper investigates the clustering problem, so all speech intervals are created based on the labeled segments from the ICSI dataset [9]. The undergone experiments show the high

quality of clusterization using the proposed algorithm.

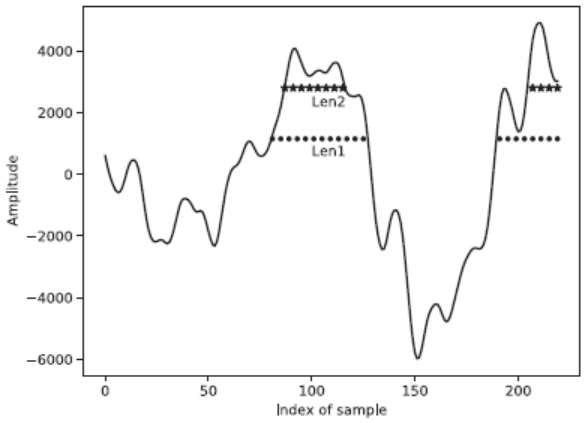

Fig.1. The base for inference of local features of the signal

While the account, we have to compare various functions and sequences constructed in the paper. For these purposes, we use the following metrics. The standard norm L_2 of the function *f(t)* with $t \in \{U,V\}$; defined according to (1):

$$||L_2(f)||^2 = \int_U^V |f(t)|^2 dt, \qquad (1)$$

and the l_2 norm of the real series $S = a_0, a, s_2, \dots, a_{N-1}$ is defined as

$$||l_2(S)||^2 = \sum_i a_i^2. \qquad (2)$$

We also need to compare a function *f(t)* with its approximation *Appr(t)*. We leverage standard signal to noise ratio (SNR) to this end:

$$SNR = 10 \cdot \log_{10}\left(\frac{\sigma^2(f(t)}{\sigma^2(f(t) - Appr(t))}\right), \qquad (3)$$

where σ^2 is variance and $||L_2(f)|| = ||L_2(Appr(t)||$. The package [7] was utilized for all calculations performed in this paper.

We often have to mark the length of a chunk used in the calculation. We use two methods: time in *ms*, if the sampling frequency is pointed out, or only as a digit. The latter means that the length is measured in samples.

The paper is organized as follows. First, we give the introduction. Some definitions and ideas about conversion speech file to step function and finding suboptimal thresholds are in the next two sections. Then we describe the application of obtained features of speech signal through step function in speaker diarization. After that, we explain the utilization of instantaneous frequency as well as experimental results and make conclusions.

II. Change Sine with the Step-Function

It follows from Fig. 1 that the first problem we have to solve is a fast optimal algorithm for choosing the levels for the construction step-function. Let *f(t)* be a function and *Appr(t)* be its step-wise version with a given number of levels. The most evident criterion for optimization is

$$||L_2(f - Appr)|| \to min, ||L_2(f)|| = ||L_2(Appr)|| \qquad (4)$$

In what follows, we write $||f||$ instead of $||L_2(f)||$.

This problem was investigated in [10] under the supposition that the source signal has a normal distribution. We suppose that *f(t)* is a regular function, and we start with the case $f(t) = \sin(K \cdot t)$ where *K* is an integer.

Proposal

The optimal levels for approximation $\sin(K \cdot t)$ by step function are independent of *K*.

Proof.

First of all, we have to select a quantum of thresholds used to create *Appr(t)*. While proving, we limit ourselves, admitting the number of thresholds equal to one and the function domain equal to $\left[0, \frac{\pi}{K}\right]$. The general case can be considered the same.

We have a single threshold Thr, and the step-function *Step* is defined according

$$Step(t) = \begin{cases} 0 & if \ \sin(K \cdot t) < Thr \\ Thr & otherwise \end{cases}.$$

Now $Appr(t) = C \cdot Step(t)$ where the constant *C* provides

$$||\sin(K \cdot t)|| = ||Appr(t)||.$$

Define $0 < A < \pi/K$ via equation $Thr = \sin(K \cdot A)$ and set $PiK = \frac{\pi}{K}$. The equivalent definition of $Appr(t)$ is

$$Appr(t) = \begin{cases} 0 & if \ t < A \ or \ t > PiK - A \\ C \cdot Thr & otherwise \end{cases}. \qquad (5)$$

Let $D = C \cdot Thr$. We have

$$||\sin(K \cdot t))||^2 = \int_0^{PiK} \sin^2(K \cdot t) dt = PiK/2, \qquad (6)$$

$$||Appr(t))||^2 = D^2(PiK - 2 \cdot A). \qquad (7)$$

To equalize the right parts of (6) and (7), we must set the value

$$D^2 = \frac{PiK}{2 \cdot (PiK - 2 \cdot A)}. \quad (8)$$

We get

$$||\sin(K \cdot t) - Appr(t))||^2 = \int_0^A \sin^2(K \cdot t)dt + \int_A^{PiK-A} (\sin(K \cdot t) - D)^2 dt + \int_{PiK-A}^{PiK} \sin^2(K \cdot t)dt = \int_0^{PiK} \sin^2(K \cdot t)dt + D^2 \cdot (PiK - 2 \cdot A) - 2 \cdot D \int_A^{PiK-A} \sin(K \cdot t)dt.$$

Thus taking into account equations (7) and (8), we get

$$||\sin(K \cdot t) - Appr(t))||^2 = PiK - 4 \cdot D \cdot \cos(K \cdot A)/K = PiK - 4 \cdot \sqrt{\frac{\pi}{2 \cdot (\pi - 2 \cdot K \cdot A)}} \cos(K \cdot A)/K. \quad (9)$$

We see that the right part of (9) is a function of $K \cdot A$ since this function attains its minimum at a point $K \cdot A_0$. This point is independent of K and the value of $Thr = \sin(K \cdot A_0)$ is also independent of K.

This statement ends the proof.

We present the final results of the calculations relating to the cases of two and three thresholds. Here

$$Dist^2(t) = ||\sin(K \cdot t) - Appr(t)||^2.$$

In the case of two thresholds,

$$Thr_1 = \sin(K \cdot A),\ Thr_2 = \sin(K \cdot B),$$

and

$$Dist^2 = PiK + 4 \cdot Coe \cdot (Thr_1 \cdot (\cos(K \cdot B) - \cos(K \cdot A)) - Thr_2 \cdot \cos(K \cdot B)),$$
$$D = 2Thr_1^2 \cdot K \cdot (B - A)$$
$$E = Thr_2^2 \cdot (\pi - 2 \cdot K \cdot B) \quad (10)$$
$$Coe^2 = \frac{\pi}{2(D + E)}.$$

In the case of three thresholds,

$$Thr_1 = \sin(K \cdot A),\ Thr_2 = \sin(K \cdot B),\ Thr_3 = \sin(K \cdot C).$$

We have

$$Dist^2 = PiK + 4 \cdot Coe \cdot (Thr_1 \cdot (\cos(K \cdot B) - \cos(K \cdot A)) + Thr_2 \cdot (\cos(K \cdot B) - \cos(K \cdot C)) - Thr_3 \cdot \cos(K \cdot C)),$$
$$D = 2 \cdot Thr_1^2 \cdot K \cdot (B - A) \quad (11)$$

$$E = 2 \cdot Thr_2^2 \cdot K \cdot (C - B) + Thr_3^2 \cdot (\pi - 2 \cdot K \cdot C)$$
$$Coe^2 = \frac{\pi}{2(D + E)}.$$

We see that $Dist(t)$ is a function of *KA, KB*, and *KC*. The optimal values of the thresholds are independent of K.

In the general case, the absolute values of the right parts of (9), (10), and (11) depend on *K*. At the same time, the evaluation in SNR terms (3) does not. We face nonlinear optimization problems and estimate the optimal values by direct calculation, where all arguments are changed with a small step. The found optimal thresholds are collected in Table I.

The presented results are based on a fast calculation of the suboptimal threshold for arbitrary fragments of speech files. Using Table I as a template, we can also evaluate the possible approximation of fragment by step-function in terms of SNR that can be gained in arbitrary speech fragments.

TABLE I. EXACT VALUES OF BEST THRESHOLDS FOR THE APPROXIMATION OF SINE FUNCTION

Number of thresholds	Thr_1	Thr_2	Thr_3	SNR (dB)
1	0.39	---	---	11
2	0.27	0.6	---	15
3	0.2	0.44	0.72	17.8

III. FAST PROCEDURE TO OBTAIN SUBOPTIMAL THRESHOLDS FOR FRAGMENTS OF SPEECH FILES

An arbitrary fragment of a speech file can be decomposed into a Fourier series; that is, a signal can be decomposed into its constituent frequencies. The previous section results can be viewed as a hint that optimal thresholds are independent of the fundamental frequency and depend only on the fragment's power. In this section, we are going to test this hypothesis. We need an algorithm to calculate thresholds since the direct calculation, according to (4), is very costly.

A. Suboptimal thresholds and k-means procedure

Supposing that the source signal is a random function with normal distribution, one can use the theory proposed in [10] for solving the problem (4). In the case of a discrete-time argument, the procedure based on *k-means* is suggested in [11]. The justification of the algorithm can be found in [10]. Recall that the *k-means* procedure creates centroids of the prescribed number *P* of clusters containing the given data. The procedure leverages the centroids' random initial values, so a series of runs of the function can lead to different results.

Nevertheless, this approach is trendy since there are no preliminary restrictions on the source data. Any item from the

source is placed in a cluster with a centroid with minimal Euclidian distance to the item. That is not precisely the problem we have to solve, so a method for creation thresholds was proposed in [11]. Algorithm 1 describes the production of the thresholds intended for a solution to our problem.

Algorithm 1 The case of three thresholds. Obtaining suboptimal values

$AFragm$ = **abs**($Fragm$);
$C_0, C_1, \ldots, C_{P-1}$ = **kmeans**($AFragm$,P);{Create centroids of P clusters}
for $I = 0$ to $P - 2$ **do**
  $Thr_I = (C_I + C_{I+1})/2$;
end for

In our case, we do not consider fragments of speech files as a stochastic process. Instead, we took the *Sine* function as a template. So we modified the formulas in Algorithm 1, trying to get the results presented in Table I using a discrete series of sinus values as a source signal. We propose empirical formulas, which are placed in Table II, that construct thresholds based on centroids. The centroids produced by *kmeans* must be sorted before applying the formulas.

TABLE II. SUBOPTIMAL THRESHOLDS BY K-MEANS PROCEDURE WITH GIVEN NUMBER P OF CLUSTERS

P	Centroids	Thr_1	Thr_2	Thr_3
2	C_0, C_1	$0{,}83C_0+0{,}17C_1$	--	--
3	C_0, C_1, C_2	$0{,}83C_0+0{,}17C_1$	C_1	--
4	C_0, C_1, C_2, C_3	$0{,}83C_0+0{,}17C_1$	C_1	C_2

We compared results obtained with the thresholds built by Algorithm 1 and the modified thresholds, according to Table II. To do this, we calculated SNRs of approximation a source file by step-functions produced by the first and by the second sets of thresholds. We divided a file into 72 fragments of length 512 and performed compare of each fragment. The graph of produced SNRs is shown in Fig. 2.

One can see that the SNRs vary significantly from one fragment to another. To get a digital value for the quality of an approximation, we use the median of all obtained SNRs. In what follows, we compare two methods using this parameter. Some results of the comparison are placed in Table III.

TABLE III. MEDIANS OF SNRS, DEPENDING ON THE METHOD OF PRODUCING THRESHOLDS

File	SNR (dB) Algorithm 1	SNR (dB) Table II
Fl1	7.2	9.3
Fl2	6.7	9.3
Fl3	7.1	10.2
Fl4	7.4	9.9
Fl5	7.2	10.0
Fl6	7.1	9.9

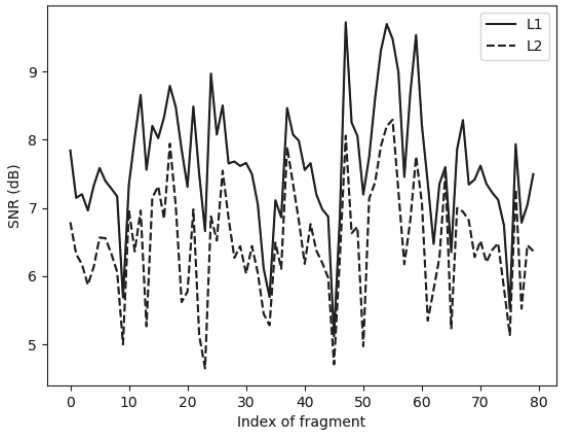

Fig. 2. SNRs of approximation source signal by step-function with different sets of thresholds; two thresholds are utilized; L1 - thresholds are calculated by modified thresholds; L2 - thresholds are calculated through Algorithm 1.

In Table I, the SNRs describing the quality of approximation of Sine by step function are shown. It is interesting to equate the obtained values with the optimal digits depending on the number of thresholds built on the base of Table II. The length of the fragment equals 256. The results are presented in Table IV.

TABLE IV. MEDIANS OF SNRS DEPENDING ON THE NUMBER OF THRESHOLDS

File	SNR (dB) 1 Thr	SNR (dB) 2 Thr	SNR (dB) 3 Thr
Fl1	6.6	9.3	11.3
Fl2	7.2	9.3	11.3
Fl3	7.1	10.2	12.1
Fl4	7.6	9.7	11.6
Fl5	6.7	9.3	10.7

It can be seen that the gained quality of approximation in terms of SNR weakly depends on file, but it significantly less than the quality pointed out in Table I. Considering the results from Table III, we conclude that there is slack dependence on the length of fragments .

B. Fast evaluation of thresholds by linear regression

This research aims to develop a procedure that fits online diarization, so most attention must be drawn to minimize calculation resources. We will implement a linear regression to obtain a formula that provides a fast calculation of thresholds based on linear regression.

For the function Q *sin,* where Q is a constant, the optimal thresholds are proportional to Q. Let us denote by *Mx* the maximum of the fragment, and by *Std* - the standard deviation of samples in the chunk. Immediate utilizing the value *Mx* as an argument for the regression is a sterile solution since this parameter changes very significantly from one chunk to another. Thus, we leverage *Std/Mx* as an argument and *Thr/Mx* as a target where *Thr* is calculated using the above-described algorithms. The value that the regression predicts is multiplied to *Mx,* and the deduced value *Pred* is the optimal threshold estimate.

One example of the formula obtained via the standard regression procedure is presented in (12). We used a speech file written with 16000*Hz* and an interval length of 16*ms* during the calculation.

$$Thr0 = Mx \cdot (0.31970 \cdot Std/Mx + 0.02123),$$
$$Thr1 = Mx \cdot (0.80885 \cdot Std/Mx + 0.02702). \quad (12)$$

A comparison of SNRs results for the same fragments obtained by applying thresholds calculated employing the *k-means* function and by thresholds obtained using linear regression are shown in Fig. 3. One can see that the data differ very weakly.

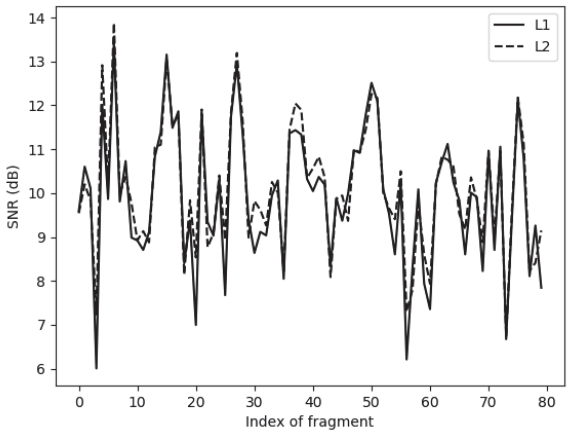

Fig. 3. SNRs of approximation source fragments of length 512 by step-function, two thresholds are utilized; L1 - thresholds are calculated by the modified procedure in Table II, L2 – through linear regression

The data collection that is required before implementing our speaker diarization procedure is a spending process. In the course of our experiments, we observed the following pattern. One could use a "general" prediction function that is good for any speech file written with the same frequency and the same length of chunks.

To prove this statement, we randomly have chosen six files from the dataset and applied two algorithms to estimate thresholds: one is the *k-means* procedure with the thresholds calculated according to Table II; the other is the estimation through (12). We created two step-functions after obtaining the thresholds and tested the source signal's approximation by these two functions. The results of the approximation in terms of SNR are presented in Table V. One can see that the difference is small, so we can calculate the regression coefficients in advance and use them for construction step functions for files under consideration. Of course, it is true if the sampling frequency and the length of the fragment are stable.

IV. Instant Frequency

We are going to use some of the features of the waveform. One is the ratio *Len2/Len1* (see Fig.1), and another feature is based on the instant frequency of the signal at point [8].

Recall the basic definitions. Let *x(t)* be a signal, *y(t)* be its Hilbert transform and

$$z(t) = x(t) + j \cdot y(t)$$ [8].

In this case, $z(t) = r(t)e^{j\omega(t)}$ is an analytic signal. In the neighborhood of a point , the value of

$$w(t) \approx w_0 + (t - t_0)w_1$$

and w_1 is the signal's instant frequency at the point t_0.

TABLE V. Medians of SNRs approximations source signals by step-functions with thresholds obtained by k-means and "general" linear regression

File	SNR (dB) k-means	SNR (dB) Regression
Fl1	9.4	9.5
Fl2	10.1	10.0
Fl3	10.3	10.2
Fl4	9.7	9.6
Fl5	9.2	9.4
Fl6	9.7	9.8

Let *{[x[n]}* be a sequence. It is known that the analytic signal can be utilized for the construction envelope of this sequence. In the general case, an approximation of Hilbert's transform of *{[x[n]}* is performed by applying a finite impulse response filter to the sequence. We used another procedure that is good just for extreme points of signal. Let *{[x[n]}* be a local maximum of the sequence. It is proved that the instant frequency at this point can be estimated through (13):

$$w_1/Fr = \arccos\left(\frac{x[n_0 - 1] + x[n_0 + 1]}{2 \cdot x[n_0]}\right) \quad (13)$$

where the number *Fr* is the sampling frequency [12]. The value w_1 defined by the formula (13) is multiplication-invariant by a constant so that it can be viewed as a waveform feature.

V. Speaker Diarization

This section develops a simple procedure based on the previous results for clusterization speakers in the diarization process. The basic idea of the clusterization is distinguishing speakers by forms of waves. This form is described by the ratio *R=Len2/Len1* (Fig 1) and the instant frequency *InFr* at the maximal point. In Fig. 4, the distribution of points with coordinates *(Fr, R)* for different speakers is shown. One can see that it is not easy to construct a discriminant function for the speakers' separation. Below, we describe the steps intended to develop such a function.

The procedure is as follows. Before the start, we need a set of files containing all the speakers' speech recordings taking part in the measure. A program creates hashes on the base of the submitted files. During the session, the same program analyzes speech, constructs hashes, and calculates distances from templates produced in advance. The current fragment points to the speaker with a minimal distance of the hash to the template.

The first problem is developing hashes invariant to the fragment's length and multiplication fragment by a constant. The average values of the waveform features fit these conditions. We introduce some auxiliary functions which simplify the description of the hash production algorithm.

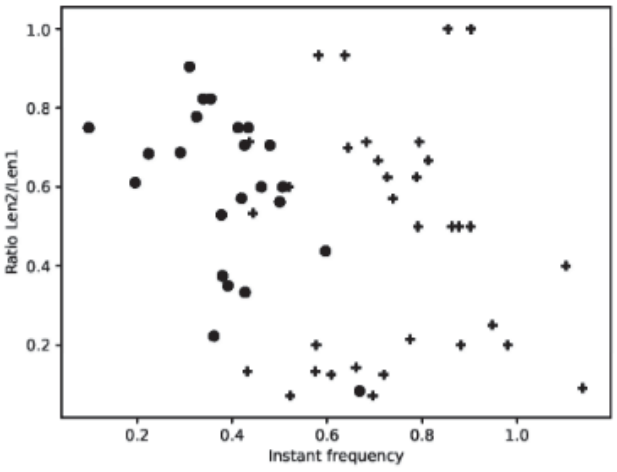

Fig. 4. Distribution of points related to two male speakers

1. ***Inst*** = ***instFr***(*Chunk)* - calculates instant frequency in the sequence *Chunk* according to (13);
2. ***ListOfPairs = stepFun***(*Chunk, Thr)* – produces the list of pairs *(Beg,End)* - beginning and end of intervals where *Chunk>Thr* ;
3. ***U,V = histo***(*Chunk, Bins*) - produces a histogram where *U* is the number of items in bins, *V* - bounds of bins, *Bins* - number of bins or bounds of bins;
4. ***Thr0,Thr1 = regr***(*Chunk, Reg1, Reg2)* - calculates a thresholds using coefficients of regression *Reg1, Reg2* via the formula of type (12);
5. ***D = dist***(*U, V)* - Euclidian distance between two vectors.

The source fragment *Frag* of the speech file is divided into chunks of the constant length of *Ln*. The algorithm consists of two parts. The first part deals with a single chunk, and the second part processes results obtained from all chunks. Each chunk is treated according to Algorithm 2. Here *Bins* is the number of bins used in the histogram function.

The chunks' processing is used to create the final hash vector of the fragment *Frag* by Algorithm 3. We separate all records into three parts, using parameter *Len1* as a key. Then we sum up all the records in each part. Finally, we concatenate three vectors, and the normalized version of the vector is used as the final hash vector of the fragment.

VI. Experiment

We use the dataset [9] as a source of speech files. All the files are written with a 16000 Hz sample frequency. Using segments in the dataset, we selected 43 files belonging to five speakers - three men and two women and duration from 1 to 2.5 seconds. The results of diarization are placed in Table VI. One can see that the results relating to Woman2 fall out of the list. We suppose that it is a consequence of the speaker's emotional state since that state changes during the speaking.

Algorithm 2 Obtaining parameters of chunk

Require: $Chunk, Reg1, Reg2, Ln, Bins$
Ensure: $Listofrecords$
  $AllRecrds = []$ { All records are collected here}
  $Thr0, Thr1 == \mathbf{regr}(Chunk, Reg1, Reg2)\{Thr0 < Thr1\}$
  $ListOfPairs = \mathbf{stepFun}(Chunk, Thr0)$
  for $Pair$ in $ListOfPairs$ **do**
    $Beg, End \leftarrow Pair$
    $Len1 = End - Beg + 1$
    $SmallChunk = Chunk(Beg, End)$ {Part of $Chunk$ between Beg and End}
    $SmallPairs = \mathbf{stepFun}(SmallChunk, Thr1)$
    $Beg, End \leftarrow SmallPairs$ {Used the firs pair}
    $Len2 = End - Beg + 1$
    $InFr = \mathbf{instFr}(SmallChunk)$
    $U, V = \mathbf{histo}(InFr, Bins)$ {Only one item in U equals 1 and other are zeros}
    $U1, V1 = \mathbf{histo}(Len2/Len1, Bins)$ {Only one item in $U1$ equals 1 and other are zeros}
    $Record \leftarrow (Len1, U, U1)$ { Create record}
    $AllRecrds \leftarrow Record$ {Add record to the list}
  end for
  return $AllRecords$

TABLE VI. Results of diarization

Owner	Number of files	Number of errors
Man1	7	1
Man2	5	1
Man3	6	0
Woman1	9	1
Woman3	16	6

VII. Conclusion

The paper presents a new method for clustering the speakers of an audio file. The step function application allows extracting the original signal's most efficient functions to solve speaker diarization. The diarization procedure requires small resources and brings acceptable results. Change of emotional state of the speaker during dialog can lead to problems.

Algorithm 3 Consruction final hash vector of fragment

Require: $Fragm, Ln$
Ensure: $HashVector$
$AFragm \leftarrow Fragm$ {Absolute value}
$ListOfChunks \leftarrow AFragm, Ln$ { Non overlapping chunks of length Ln}
$Zer0, Zer1, Zer2 \leftarrow 0$ {Zero vectors having length of records minus 1}
$AllRecrds = []$ { All records produced by chunks are collected here}
for $Chunk$ in $ListOfChunks$ **do**
  $LocalRecord \leftarrow Chunk$
  $AllRecrdsRecordLocalRecord$ {Add records}
end for
$AllLengths \leftarrow AllRecrds$ {Collect all parameter $Len1$ from all records}
$U, V = \text{histo}(AllLengts, 3)$ {V[0],V[1],V[2],V[3]} $--boundsofbins$
for $Record$ in $AllRecrds$ **do**
  $Len1 \leftarrow Record$
  $R \leftarrow Record$ {Remove $Len1$ from $Record$}
  if $Len1 < V[1]$ **then**
    $Zer0 = Zer0 + R$
  else if $Len1 > V[2]$ **then**
    $Zer2 = Zer2 + R$
  else
    $Zer1 = Zer1 + R$
  end if
end for
$HashVector = (Zer0, Zer1, Zer2)$
$normalize(HashVector)$
return$HashVector$

ACKNOWLEDGMENT

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University .

REFERENCES

[1] S. Tranter and D. Reynolds, “An overview of automatic speaker diarization systems,” IEEE Trans. Audio, Speech, and Lang. Proc., vol.14, no.5, pp.1557-1565, Sep. 2006.
[2] Q. Wang, C. Downey, L. Wan, P. Mansfield, I. Moreno, “Speaker diarization with LSTM,”in Proc. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Apr. 2018.
[3] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, Ch. Wang, “Fully supervised speaker diarization,” in Proc.2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 2019.
[4] Y. Fujita, N. Kanda, S. Horiguchi, A. Nagamatsu, S.Watanabe, “End-to-end neural speaker diarization with permutation-free objectives,” in Proc. INTERSPEECH 2019, Sep. 2019.
[5] J. Roth et al, “ AVA-ActiveSpeaker: An audio-visual dataset for active speaker detection,” in Proc. 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), May 2020.
[6] N. Zeghidour and D.Grangier, “Wavesplit: End-to-end speech separation by speaker clustering,” in arXiv:2002.08933v2 [eess.AS], Jul 2020.
[7] F. Pedregosa et al., “Scikit-learn: machine learning in Python,” Journal of Machine Learning Research, vol.12, pp. 2825-2830, 2011
[8] F.W.King, Hilbert transform . Cambridge: Cambridge University Press, 2009
[9] ICSI Corpus, Web: \url{http://groups.inf.ed.ac.uk/ami/icsi/}
[10] S. Lloyd, “Least squares quantization in PCM”, IEEE Trans. Inform. Theory, vol IT-28, pp. 129-136, 1982.
[11] B. Girod, B., “Image and Video Compression,” Web: \url{https://web.stanford.edu/class/ee398a/handouts/lectures/05-Quantization.
[12] R. Latypov, R. Nigmatullin, E. Stolov, “Instantaneous frequency and detection of dynamics in speech, in Proc. 2017 International Symposium ELMAR, Sept. 2017.