INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

An Approach to Online Speaker Change Point Detection Using DNNs and WFSTs

Lukas Mateju, Petr Cerva, Jindrich Zdansky

Faculty of Mechatronics, Informatics and Interdisciplinary Studies,
Technical University of Liberec, Studentska 2, 461 17 Liberec, Czech Republic
{lukas.mateju, petr.cerva, jindrich.zdansky}@tul.cz

Abstract

In this paper, a new approach to speaker change point (SCP) detection is presented. This method is suitable for online applications (e.g., real-time broadcast monitoring). It is designed in a series of consecutive experiments, aiming at quality of detection as well as low latency. The resulting scheme utilizes a convolution neural network (CNN), whose output is smoothed by a decoder. The CNN is trained using data complemented by artificial examples to reduce different types of errors, and the decoder is based on a weighted finite state transducer (WFST) with the forced length of the transition model. Results obtained on data taken from the COST278 database show that our online approach yields results comparable with an offline multi-pass LIUM toolkit while operating online with a low latency.

Index Terms: speaker change point detection, deep neural networks, convolutional neural networks, real-time processing, weighted finite state transducers

1. Introduction

The SCP detection (often called speaker segmentation) is a task of determining precise change points between two heterogeneous speakers in the input utterance. This task is usually performed without any prior knowledge of the identity or even the number of speakers (i.e., it is treated as speaker independent). In practice, it is usually preceded by speech activity detection (SAD), whose goal is to filter out non-speech segments at first. Therefore, a typical SCP module provides speaker homogeneous speech segments on the output and, in conjunction with a speaker clustering module, forms an integral part of a majority of speaker diarization systems. These systems may be utilized in a wide range of speech processing applications, such as broadcast news transcription, audio indexing or data retrieval.

2. Related work

The SCP detection may be carried out using various types of input features. In the beginning years, more straightforward ones, such as zero-crossing rate or pitch [1], were successfully employed. Mel-frequency cepstral coefficients (MFCCs) [2, 3] were probably the most commonly used features, followed by line spectrum pairs [4]. Recently, the main focus has shifted to crafting more complex features capturing more speaker-specific information. Nowadays, i-vectors [5, 6] are the go-to features for most state-of-the-art systems. Alternatively, deep neural networks (DNNs) have also been successfully utilized to extract complex features [7, 8]. Furthermore, d-vectors were presented in [9], yielding excellent results. The latest trend goes in the direction of deep speaker embeddings [10, 11] designed for end-to-end systems. In practice, the best results are often achieved by a combination of the features mentioned above.

The SCP detection approaches can be divided into three main categories: metric-based, model-based and hybrid-based. The former approaches require a distance metric to be defined first. After that, two adjacent windows are shifted alongside the recording, and the distance between them is computed. If the distance is greater than a predefined threshold (fine-tuning is the main issue), a change point is detected. The most commonly used distance metrics include the Bayesian information criterion (BIC) [12, 13], the generalized likelihood ratio [14] or the Kullback-Leibler divergence [15]. The model-based approaches utilize trained models from labeled audio data to detect speaker change points. Among the most common approaches, there are the hidden Markov models [16], the Gaussian mixture models (GMM) [17], and the eigenvoice-based models [18]. The recent advances in deep learning have brought new approaches based on DNNs [19, 8], CNNs [20, 21], unidirectional [22] or bidirectional [23, 24] long short-term memory recurrent neural networks, all yielding state-of-the-art results.

Speaker change points can be detected in a) offline as well as in b) online mode. In the former case, real-time processing and low latency are not crucial. On the contrary, both these restrictions are critical in approaches suitable for the latter mode. Moreover, an online SCP module may perform only one left-to-right pass through the input data. Most of the approaches cited so far were designed with regard to the best possible quality of detection, and all of them are, of course, applicable to offline processing. However, the previously mentioned restrictions are usually not taken into account during design, and the usability of these methods for online mode is therefore limited (or not discussed in the respective papers). That means that the number of approaches explicitly designed for real-time processing (e.g., of broadcast news) is limited.

In the early stages, an online two-step SCP detector utilizing the Bayesian fusion method for fusing multiple features was proposed [25, 26]. Other works focused on XBIC [27], GMMs [17, 28, 29] or GMM-UBM [30]. In [31], the authors explored BIC, i-vectors and within class covariance normalization. The use of i-vectors was also investigated in [32]. Features extracted from DNN were explored in [33]. Finally, the authors in [34] studied in detail the influence of the online environment on various SCP detection approaches to diarization systems.

In this paper, our one-pass approach to online SCP detection is presented. This approach is developed in a series of experiments (described in Sect. 4), where various types of features, artificial training data, different DNN architectures, and several decoders with varied transduction models are investigated with respect not only to a high quality of detection but also to low latency. It should also be noted that we assume the input to our SCP detector has been pre-processed by SAD. In practice, we utilize an online SAD module described in [35].

http://dx.doi.org/10.21437/Interspeech.2019-1407

3. Evaluation metrics

Within this work, precision (P), recall (R), F-measure (F) and $\delta_{2/3}$ are used to measure the quality of SCP detection. All these values can be expressed given the alignment between the detected and reference change points [36]. Note that the measure $\delta_{2/3}$ gives (in seconds) the maximum error of the alignment for the first two-thirds of the sorted (best) hits. Another two metrics, real-time factor (RTF) and latency (L), have been employed to monitor the performance from a real-time processing point of view. The former metric[1] is defined as a ratio of processing time to the duration of the input recording. The latter one represents an average time between the actual change point and the moment the decoder outputs the corresponding label.

4. The proposed SCP approach

4.1. Data used

For training, 20,000 recordings, each with an average length of 5 seconds, have been prepared with the help of automatic Czech TV/radio broadcast data transcriptions. Each of these recordings contains exactly one speaker change point (i.e., the set consists of 20,000 speaker transitions). These transitions can be divided into four distinct groups (female to female, female to male, male to female, and male to male). Each of them is represented by 5,000 change points.

The annotations of this data (for SCP detection) are generated in a fully automated way. The frame corresponding to the actual change point, as well as the safety collar frames around it, are labeled as change points. This safety collar is set to 1 second (100 frames), i.e., 50 frames before and 50 frames after the actual change point are considered as speaker transition frames. That is due to the fact that a) determining the precise change point is quite often an ambiguous task (silence, crosstalk, etc.), and b) it is necessary to provide DNN training with enough information about the speaker transitions. The remaining frames are labeled as no change point. An example of annotation of one recording is shown in Fig. 1.

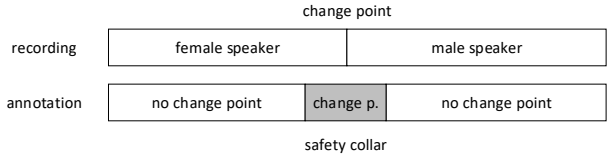

Figure 1: *An example of annotation of training data.*

For development purposes, the Czech train subset of standardized COST278 [37] pan-European broadcast news database has been utilized. Precise annotations are provided by the database. For evaluation, the Czech test subset of COST278 is employed. It consists of four recordings of different Czech broadcasts in a total length of 90 minutes. It contains not only clean speech segments but also segments with background noise and jingles. In total, 399 speaker change points are labeled within the data.

4.2. Reference results

To obtain reference results with an offline system, publicly available LIUM speaker diarization toolkit [38] has been used. The SCP portion of the system is covered by BIC segmentation and BIC clustering, followed by segmentation based on Viterbi decoding and boundary adjustments. The system is also supplemented with a pre-trained model fine-tuned for TV and radio broadcasts (i.e., our target task). During the evaluation, the LIUM toolkit has been operated with an RTF of 0.016, achieving reference results in F-measure of 84.6% and $\delta_{2/3}$ of 0.13 seconds (see the first row in Table 1 for more detailed results).

4.3. Initial approach based on DNN and WFST

The initial SCP detection approach we have developed is based on DNN trained as a binary classifier (change point/no change point) and WFST designed as an online decoder, detecting speaker transitions given the output from DNN. It closely follows an online SAD approach we proposed in [35].

The binary deep neural network has been trained using the following hyper-parameters: 2 hidden layers with 64 neurons per layer, the ReLU activation function, a learning rate of 0.08, mini-batches of size 1024, and 15 epochs. 39-dimensional MFCCs have been employed for the feature extraction. The input feature vector is formed by concatenating 100 previous frames, the current frame and 100 following frames (i.e., a 2-second context window). No local normalization has been applied. Note that all DNNs (i.e., for all experiments) are trained on GPU using the PyTorch framework[2].

As stated above, the WFSTs are utilized (using the OpenFst library[3]) as an online decoder. The decoding scheme consists of two transducers (see Fig. 2). The upper one models the input signal, while the lower one is the transduction model and represents the change point detection. It consists of two states, 0 and 1. The transitions between states 0/1 emit labels the start/end change points. The resulting change point is placed in the middle between these two labels. The transitions are also penalized by factors P1 and P2, whose values have been fine-tuned on the development set.

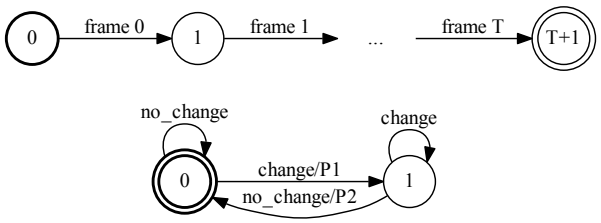

Figure 2: *Transducers representing the input signal (upper) and the transduction model (lower).*

Given the two transducers described above, the decoding process is performed using on the fly composition of the transduction and the input model of an unknown size. That method is possible since the input is considered to be a linear-topology, unweighted, epsilon-free acceptor. After each composition step, the shortest-path (considering the tropical semiring) determined in the resulting model is compared with all other alternative hypotheses. When a common path (i.e., with the same output label) is found among these hypotheses, the corresponding concatenated output labels are marked as the resulting fixed output. Since the remaining portion of the best path is not known with certainty, it is denoted as a temporary output (i.e., it can be further refined).

[1] measured by processor Intel Core i7-3770K @ 3.50GHz

[2] https://pytorch.org/

[3] http://www.openfst.org/twiki/bin/view/FST/WebHome

Table 1: *A comparison of investigated SCP detectors on COST278 Czech test subset.*

approach	precision [%]	recall [%]	F-measure [%]	$\delta_{2/3}[s]$	RTF	latency $[s]$
LIUM toolkit	**89.9**	80.0	**84.6**	**0.13**	**0.016**	-
DNN + WFST decoder	59.4	63.6	61.4	0.24	0.022	2.4
+ enhanced data	67.0	70.7	68.8	0.21	0.022	2.3
+ Δ MFCC	72.8	74.7	73.7	0.19	0.024	**1.9**
CNN	79.3	77.8	78.6	0.17	0.054	**1.9**
+ 2.5 s context window	80.3	**81.8**	81.1	0.17	0.054	2.3
+ 1 s long transition model	82.7	**81.8**	82.2	0.17	0.065	2.9

The results are presented in the second row of Table 1. They show that the decoder is capable of operating in real-time with an RTF value of 0.022. This approach, combined with the latency of 2.4 seconds, allows it to be seamlessly used in an online environment. Although the achieved results provide a decent starting point, the precision is particularly weak and overshadowed by LIUM toolkit (i.e., 59.4% vs. 89.9%). Therefore, our next goal is to improve the quality of the SCP detection.

4.4. Enhanced training set

After thoroughly evaluating the results obtained so far, two types of errors are the most prominent. The first one is represented by change points omitted due to the quick artificial transitions between speakers (e.g., director cuts in broadcast news) while the second type results in change points falsely detected because of a silence longer than 0.5 seconds in speaker homogeneous segments (caused by deep breathing or hesitation).

As a solution to the first issue, 10 hours of recordings have been prepared by artificially joining utterances of two different speakers. In total, 14,340 change points with a uniform distribution between all transition types (female-female, female-male, etc.) have thus been added to the DNN training set. To reduce the latter type of errors, another 10 hours of additional training data have been prepared. This data focuses on speaker homogeneous segments with frequent occurrences of long silences.

The results gathered in the third row of Table 1 show that the use of enhanced training data set leads to significant improvement in all of the evaluation metrics observed. For example, the F-measure value gets boosted up from 61.4% to 68.8%, while $\delta_{2/3}$ is enhanced to 0.21 seconds. Additionally, the average latency has been slightly reduced, namely, from 2.4 seconds to 2.3 seconds.

4.5. Acoustic features

In the next experiments, several feature extraction techniques are explored. In addition to the 39-dimensional MFCCs, we have also utilized 13-dimensional MFCCs with Δ and $\Delta\Delta$ coefficients (i.e., a 39-dimensional feature vector as well), and 39-dimensional bottleneck features (BTNs) extracted from the DNN trained for speech recognition (as suggested for the speaker and language identification, e.g., in [39]). Detailed information about our BTN feature extractor can be found in [40].

The results obtained are shown in Table 2. They show that the BTN features have yielded significantly worse results in all of the observed metrics (e.g., the F-measure value has dropped from 68.8% to 56.7%). On the contrary, the MFCCs with the Δ and $\Delta\Delta$ coefficients outperform the originally chosen MFCC configuration. Both the quality and the real-time performance of the SCP detection have been improved (e.g., the latency is reduced from 2.3 seconds to 1.9 seconds because the decoder is able to make the final decisions more rapidly). A likely reason is the additional information provided by the Δ and $\Delta\Delta$ coefficients.

Table 2: *A comparison of various feature extraction techniques.*

features	P	R	F [%]	$\delta_{2/3}[s]$	RTF	L $[s]$
MFCCs	67.0	70.7	68.8	0.21	**0.022**	2.3
+ Δ	**72.8**	**74.7**	**73.7**	**0.19**	0.024	**1.9**
BTNs	53.7	60.1	56.7	0.26	0.070	2.9

4.6. Convolutional neural networks

In the next step, more complex NN architecture - CNNs - are investigated. This architecture has been employed for its feature representation and modeling capabilities. The utilized CNN is composed of two convolutional and two fully connected layers. The inputs consist of 201 feature maps (i.e., 2-second context windows as before) in size of 39×1. The first convolutional layer is comprised of 105 feature maps at a size of 39×1, followed by a 3:1 max-pooling layer; the second one is composed of 157 feature maps at a size of 13×1. The rest of the hyperparameters is set as stated in Sect. 4.3.

The results are summarized in the fifth row of Table 1. The utilization of the CNNs yields an overall improvement in all quality detection metrics (e.g., the F-measure value has increased from 73.7% to 78.6%). The latency remains constant while the deterioration in the RTF could be considered negligible (i.e., it is still significantly smaller than 1).

4.7. Context window size

The next experiments focus on the size of the input feature window. This additional context should result in a higher quality of the SCP detection at a cost of worse latency. Initially, we have chosen 2-second window (with 100 preceding frames, a current frame, and 100 following frames). In this part, we explore the sizes ranging from 1 second up to 4 seconds.

The results are in Table 3. As expected, the performance (i.e., F-measure and $\delta_{2/3}$) has been further improved with the additional context. On the contrary, the latency of the system is worsened with more context information by up to 2 seconds.

4.8. The WFST with a forced length of the transition model

In the last series of experiments, our aim is to further improve the results achieved so far by introducing the WFST with a forced transition model. This model is designed to reflect the annotation style of the training data. As stated in Sect. 4.1, a 1-second (100 frames) window around the actual change point

Table 3: *A comparison of different context window sizes.*

context	P	R	F [%]	$\delta_{2/3}[s]$	RTF	L $[s]$
1 s	71.3	69.4	70.3	0.21	**0.053**	**1.4**
1.5 s	71.0	72.8	71.9	**0.14**	**0.053**	1.7
2 s	79.3	77.8	78.6	0.17	0.054	1.9
2.5 s	80.3	81.8	81.1	0.17	0.054	2.3
3 s	80.0	**83.1**	81.5	0.17	0.054	2.6
3.5 s	**80.5**	82.6	81.5	0.16	0.055	3.1
4 s	80.4	**83.1**	**81.7**	0.16	0.055	3.5

is labeled as speaker transition frames. However, during the decoding, the real duration of the transition between two speakers substantially varies.

Therefore, in this experiment, the duration of the transition is forced to be exactly 1 second at first. For this purpose, the transduction model has been modified (see in Fig. 3) to correspond to the duration of the forced transition: it consists of two main states (0 and 1) and 98 transition states (shown as . . .).

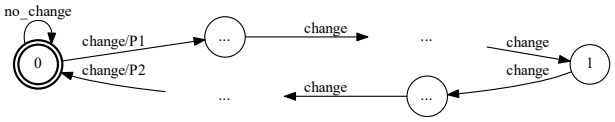

Figure 3: *A transduction model with forced transition.*

This scheme works as follows: when a speaker change occurs, the decoder moves frame by frame from state 0 through half of the transition states to state 1. Here, a new change point label is provided, and the decoder moves backward to state 0, where it waits until the next change occurs. Note that, during this process, the penalty factors P1 and P2 (tuned on the development set) are in place as well.

The results are summarized in Table 4. First, a CNN with a context size of 2.5 seconds has been used. Next, we evaluate not only the forced length of the transition at 1 second but also several other values in a range from 0.5 up to 2 seconds. The results show two contradictory trends: the quality of detection increase with the additional duration, while the RTF and latency values are worsened. Therefore, the optimal value of the duration strongly depends on the target application.

For example, with the forced length of 1 second and total latency below 3 seconds, the proposed approach still allows for performing speaker segmentation with an accuracy level approaching the offline reference system (see the last row of Table 1). If the latency is not a concern, it is possible to tune this approach to even outperform the reference system. For instance, a system based on the CNN, the context window size of 3 seconds, and the WFST with a forced length of 2 seconds yields an F-measure value of 85.6% and a $\delta_{2/3}$ value of 0.18 seconds (with the latency at 4.8 seconds).

Table 4: *A comparison of different forced lengths of the WFST.*

length	P	R	F [%]	$\delta_{2/3}[s]$	RTF	L $[s]$
0.5 s	77.2	75.2	76.2	**0.13**	**0.057**	**2.2**
1 s	82.7	**81.8**	82.2	0.17	0.065	2.9
1.5 s	83.5	81.5	82.5	0.16	0.072	3.7
2 s	**84.2**	81.5	**82.8**	0.17	0.079	4.5

5. Evaluation on full COST278 database

As a final experiment, we have only trained our system on the training subset of the COST278 database. That means that we have used the MFCCs with the Δ and $\Delta\Delta$ coefficients, the CNN instead of the feed-forward DNN, an extended context size of 2.5 seconds, and a WFST based decoder with a 1-second forced transition (the enhanced training set has not been utilized). After that, we evaluate the performance of this system on all 11 languages of the COST278 test subset and compare the results with the LIUM toolkit. Our goal has been to see if our single-pass approach (without clustering) can compete with a reference offline tool.

The results show that both approaches perform on a relatively similar level. The LIUM toolkit yields an F-measure value of 73.5% and a $\delta_{2/3}$ value of 0.21 seconds, while our approach has scored an F-measure value of 73.1% and a $\delta_{2/3}$ value of 0.15 seconds, with the latency at 2.9 seconds. Figure 4 depicts the detailed results for all COST278 languages. The easiest ones were four closely related Slavic languages – Czech, Slovenian, Croatian and Slovak. Basque and Spanish for the LIUM toolkit and Belgian Dutch and Basque for our SCP detection approach have been the most difficult instances.

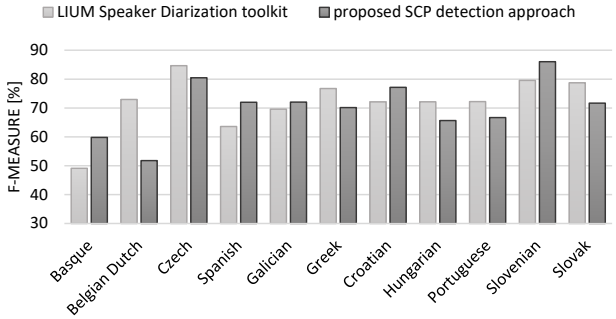

Figure 4: *A comparison of LIUM toolkit and our SCP module.*

6. Conclusions

In this paper, we have proposed a highly tunable one-pass approach suitable for both offline and online SCP detection (i.e., in our case, we treat online processing as a special case of an offline application). We showed that the settings, including but not limited to the architecture of the DNN, the type of the decoder (and its transition model), and the feature extraction technique or the context window size all have a significant impact not only on the quality of the SCP detection but also on the real-time processing capabilities of the final system. As shown by the results, our fine-tuned system for online processing is capable of operating in real-time with the latency just under 3 seconds, and at the same time, it yields results comparable to the offline reference system. On the contrary, it is possible to design an entirely offline system by tuning the settings for the best quality of the SCP detection while disregarding the latency (and RTF) value. The results of such systems could be further greatly improved by additional processing, e.g., by clustering.

7. Acknowledgements

This work was supported by the Technology Agency of the Czech Republic (Project No. TH03010018), and by the Student Grant Scheme 2019 of the Technical University in Liberec.

8. References

[1] D. Wang, L. Lu, and H. Zhang, "Speech segmentation without speech recognition," in *ICASSP 2003, Hong Kong*, 2003, pp. 468–471.

[2] R. Sinha, S. E. Tranter, M. J. F. Gales, and P. C. Woodland, "The cambridge university march 2005 speaker diarisation system," in *Interspeech 2005, Lisbon, Portugal*, 2005, pp. 2437–2440.

[3] S. Meignier, D. Moraru, C. Fredouille, J. Bonastre, and L. Besacier, "Step-by-step and integrated approaches in broadcast news speaker diarization," *Computer Speech & Language*, vol. 20, no. 2-3, pp. 303–330, 2006.

[4] L. Lu, H. Zhang, and H. Jiang, "Content analysis for audio classification and segmentation," *IEEE Trans. Speech and Audio Processing*, vol. 10, no. 7, pp. 504–516, 2002.

[5] B. Desplanques, K. Demuynck, and J. Martens, "Factor analysis for speaker segmentation and improved speaker diarization," in *Interspeech 2015, Dresden, Germany*, 2015, pp. 3081–3085.

[6] L. V. Neri, H. N. B. Pinheiro, T. I. Ren, G. D. C. Cavalcanti, and A. G. Adami, "Speaker segmentation using i-vector in meetings domain," in *ICASSP 2017, New Orleans, LA, USA*, 2017, pp. 5455–5459.

[7] K. Chen and A. Salman, "Learning speaker-specific characteristics with a deep neural architecture," *IEEE Trans. Neural Networks*, vol. 22, no. 11, pp. 1744–1756, 2011.

[8] A. Sarkar, S. Dasgupta, S. K. Naskar, and S. Bandyopadhyay, "Says who? deep learning models for joint speech recognition, segmentation and diarization," in *ICASSP 2018, Calgary, AB, Canada*, 2018, pp. 5229–5233.

[9] R. Wang, M. Gu, L. Li, M. Xu, and T. F. Zheng, "Speaker segmentation using deep speaker vectors for fast speaker change scenarios," in *ICASSP 2017, New Orleans, LA, USA*, 2017, pp. 5420–5424.

[10] H. Bredin, "Tristounet: Triplet loss for speaker turn embedding," in *ICASSP 2017, New Orleans, LA, USA*, 2017, pp. 5430–5434.

[11] A. Jati and P. G. Georgiou, "Speaker2vec: Unsupervised learning and adaptation of a speaker manifold using deep neural networks with an evaluation on speaker segmentation," in *Interspeech 2017, Stockholm, Sweden*, 2017, pp. 3567–3571.

[12] S. S. Chen and P. S. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the bayesian information criterion," in *DARPA Broadcast News Transcription and Understanding Workshop*, 1998, pp. 127–132.

[13] M. Cettolo, M. Vescovi, and R. Rizzi, "Evaluation of bic-based algorithms for audio segmentation," *Computer Speech & Language*, vol. 19, no. 2, pp. 147–170, 2005.

[14] H. Gish, M. H. Siu, and R. Rohlicek, "Segregation of speakers for speech recognition and speaker identification," in *ICASSP 1991, Toronto, Ontario, Canada*, 1991, pp. 873–876.

[15] M. A. Siegler, U. Jain, B. Raj, and R. M. Stern, "Automatic segmentation, classification and clustering of broadcast news audio," in *DARPA Speech Recognition Workshop*, 1997, pp. 97–99.

[16] S. Meignier, J. Bonastre, and S. Igounet, "E-HMM approach for learning and adapting sound models for speaker indexing," in *Speaker Odyssey, Crete, Greece*, 2001, pp. 175–180.

[17] A. S. Malegaonkar, A. M. Ariyaeeinia, and P. Sivakumaran, "Efficient speaker change detection using adapted gaussian mixture models," *IEEE Trans. Audio, Speech & Language Processing*, vol. 15, no. 6, pp. 1859–1869, 2007.

[18] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair, "Stream-based speaker segmentation using speaker factors and eigenvoices," in *ICASSP 2008, Las Vegas, NV, USA*, 2008, pp. 4133–4136.

[19] V. Gupta, "Speaker change point detection using deep neural nets," in *ICASSP 2015, South Brisbane, Queensland, Australia*, 2015, pp. 4420–4424.

[20] M. Hruz and M. Kunesova, "Convolutional neural network in the task of speaker change detection," in *SPECOM 2016, Budapest, Hungary*, 2016, pp. 191–198.

[21] M. Hruz and Z. Zajic, "Convolutional neural network for speaker change detection in telephone speaker diarization system," in *ICASSP 2017, New Orleans, LA, USA*, 2017, pp. 4945–4949.

[22] M. India, J. A. R. Fonollosa, and J. Hernando, "LSTM neural network-based speaker segmentation using acoustic and language modelling," in *Interspeech 2017, Stockholm, Sweden*, 2017, pp. 2834–2838.

[23] R. Yin, H. Bredin, and C. Barras, "Speaker change detection in broadcast TV using bidirectional long short-term memory networks," in *Interspeech 2017, Stockholm, Sweden*, 2017, pp. 3827–3831.

[24] M. Hruz and M. Hlavac, "LSTM neural network for speaker change detection in telephone conversations," in *SPECOM 2018, Leipzig, Germany*, 2018, pp. 226–233.

[25] L. Lu and H. Zhang, "Speaker change detection and tracking in real-time news broadcasting analysis," in *ACMMM 2002, Juan les Pins, France*, 2002, pp. 602–610.

[26] L. Lu and H. Zhang, "Unsupervised speaker segmentation and tracking in real-time audio content analysis," *Multimedia Systems*, vol. 10, no. 4, pp. 332–343, 2005.

[27] X. Anguera, "Xbic: Real-time cross probabilities measure for speaker segmentation," *ICSI*, pp. 1–8, 2005.

[28] K. Markov and S. Nakamura, "Never-ending learning system for on-line speaker diarization," in *ASRU 2007, Kyoto, Japan*, 2007, pp. 699–704.

[29] J. T. Geiger, F. Wallhoff, and G. Rigoll, "GMM-UBM based open-set online speaker diarization," in *Interspeech 2010, Makuhari, Chiba, Japan*, 2010, pp. 2330–2333.

[30] T. Wu, L. Lu, K. Chen, and H. Zhang, "Universal background models for real-time speaker change detection," in *MMM 2003, Taiwan*, 2003, pp. 135–149.

[31] D. Dimitriadis and P. Fousek, "Developing on-line speaker diarization system," in *Interspeech 2017, Stockholm, Sweden*, 2017, pp. 2739–2743.

[32] W. Zhu and J. W. Pelecanos, "Online speaker diarization using adapted i-vector transforms," in *ICASSP 2016, Shanghai, China*, 2016, pp. 5045–5049.

[33] Z. Ge, A. N. Iyer, S. Cheluvaraja, and A. Ganapathiraju, "Speaker change detection using features through a neural network speaker classifier," *CoRR*, vol. abs/1702.02285, pp. 1111–1116, 2017.

[34] M. Kunesova, Z. Zajic, and V. Radova, "Experiments with segmentation in an online speaker diarization system," in *TSD 2017, Prague, Czech Republic*, 2017, pp. 429–437.

[35] L. Mateju, P. Cerva, J. Zdansky, and J. Malek, "Speech activity detection in online broadcast transcription using deep neural networks and weighted finite state transducers," in *ICASSP 2017, New Orleans, LA, USA*, 2017, pp. 5460–5464.

[36] O. J. Rasanen, U. K. Laine, and T. Altosaar, "An improved speech segmentation quality measure: the r-value," in *Interspeech 2009, Brighton, United Kingdom*, 2009, pp. 1851–1854.

[37] A. Vandecatseye, J. Martens, J. P. Neto, H. Meinedo, C. Garcia-Mateo, J. Dieguez-Tirado, F. Mihelic, J. Zibert, J. Nouza, P. David, M. Pleva, A. Cizmar, H. Papageorgiou, and C. Alexandris, "The COST278 pan-european broadcast news database," in *LREC 2004, Lisbon, Portugal*, 2004, pp. 873–876.

[38] M. Rouvier, G. Dupuy, P. Gay, E. el Khoury, T. Merlin, and S. Meignier, "An open-source state-of-the-art toolbox for broadcast news diarization," in *Interspeech 2013, Lyon, France*, 2013, pp. 1477–1481.

[39] F. Richardson, D. A. Reynolds, and N. Dehak, "A unified deep neural network for speaker and language recognition," in *Interspeech 2015, Dresden, Germany*, 2015, pp. 1146–1150.

[40] L. Mateju, P. Cerva, J. Zdansky, and R. Safarik, "Using deep neural networks for identification of slavic languages from acoustic signal," in *Interspeech 2018, Hyderabad, India*, 2018, pp. 1803–1807.