Interspeech 2022
18-22 September 2022, Incheon, Korea

Token-level Speaker Change Detection Using Speaker Difference and Speech Content via Continuous Integrate-and-fire

Zhiyun Fan[1,2], *Zhenlin Liang*[3], *Linhao Dong*[3], *Yi Liu*[3], *Shiyu Zhou*[1], *Meng Cai*[3], *Jun Zhang*[3], *Zejun Ma*[3], *Bo Xu*[1]

[1]Institute of Automation, Chinese Academy of Sciences, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, China
[3]Bytedance AI LAB

fanzhiyun2017@ia.ac.cn, {liangzhenlin.lzl, donglinhao}@bytedance.com

Abstract

In multi-talker scenarios such as meetings and conversations, speech processing systems are usually required to segment the audio and then transcribe each segmentation. These two stages are addressed separately by speaker change detection (SCD) and automatic speech recognition (ASR). Most previous SCD systems rely solely on speaker information and ignore the importance of speech content. In this paper, we propose a novel SCD system that considers both cues of speaker difference and speech content. These two cues are converted into token-level representations by the continuous integrate-and-fire (CIF) mechanism and then combined for detecting speaker changes on the token acoustic boundaries. We evaluate the performance of our approach on a public real-recorded meeting dataset, AISHELL-4. The experiment results show that our method outperforms a competitive frame-level baseline system by 2.45% equal coverage-purity (ECP). In addition, we demonstrate the importance of speech content and speaker difference to the SCD task, and the advantages of conducting SCD on the token acoustic boundaries compared with conducting SCD frame by frame.
Index Terms: speaker change detection, automatic speech recognition, continuous integrate-and-fire

1. Introduction

Speaker change detection (SCD) is the task of locating the time points when a different speaker starts to speak. The SCD system is often applied as a submodule of speaker diarization, which determines "who spoke when" or is used as a front-end of automatic speech recognition (ASR). The performance of SCD will greatly affect the subsequent models.

Many different SCD systems have been proposed in the literatures. The distance-based methods [1–5] are studied earliest, which calculate the distance between features in the adjacent windows, and once the distance exceeds the threshold, a speaker change is detected. In addition, the model-based methods [6–9] segment the input audio into fix-length segments that are assumed to contain only one speaker, and models are trained to extract speaker embeddings for each segment. Then the distance between adjacent speaker embeddings is used to decide whether speaker change happens. More recently, some end-to-end SCD systems [10–13] are proposed to predict the speaker change at the end of the neural network without relying on a distance metric. Yin et al. address the SCD as a sequence labeling task, which predicts each frame as a change point or not [10]. Leda et al. extract speaker embeddings for adjacent segments with a shared siamese network, then a fully connected (FC) classifier decides whether there is a speaker change between these two segments [11]. Marek et al. propose a Convolutional Neural Networks (CNN) based system to the SCD task [13].

The SCD systems mentioned above only rely on acoustic information such as pitch [14], speaker embeddings [11], and silence [15]. However, the speech content is another critical cue for the SCD task [16]. There has been little effort in exploiting speech content for the SCD task. Meng et al. formulate text-based SCD as a binary sentence-pair classification problem, which judges whether the speaker is changing between consecutive sentence pairs [17]. Jin et al. investigate the way to integrate content and acoustic cues with sequence to sequence model to improve speaker diarization performance [18]. India et al. prove that the combination of content and acoustic features outperforms the cosine distance-based baseline where only acoustic data is used [19]. Anidjar et al. propose a hybrid framework for the SCD problem that is learned by content information and speech signals [20]. Most of the existing methods combining speaker and content information rely on the acoustic boundaries from the manual annotation or the forced-alignment results produced by the ASR, which leads to complex processes and time-consuming annotations.

In this paper, we propose an end-to-end SCD system that combines speech content and speaker difference to detect speaker changes between tokens. We construct a joint model for ASR, speaker identification (SID) and SCD. The ASR part follows the continuous integrate-and-fire (CIF) based encoder-decoder [21] structure, which can generate speech representations and token acoustic boundaries. The SID part uses the located token acoustic boundaries by the CIF to generate token-level speaker embeddings. The SCD part calculates the speaker difference between the token-level speaker embeddings and then combines speaker difference with speech content to detect speaker change. We train and evaluate the proposed model on a public real-recorded dataset, AISHELL-4. The experiment results of our method show a performance improvement compared to a competitive open-sourced frame-level baseline. Further experiments demonstrate the advantages of using speaker difference and speech content for the SCD task.

2. CIF-based ASR

The continuous integrate-and-fire (CIF) based ASR follows an encoder-decoder framework connected by the CIF mechanism, a soft and monotonic alignment [21]. The blue part in Fig.1 illustrates the structure of the CIF-based ASR. The encoder trans-

This work is supported by the National Innovation 2030 Major S&T Project of China under Grant No.2020AAA0104202 and the Key Research Program of the Chinese Academy of Sciences under Grant No.ZDBS-SSW-JSC006 and Strategic Priority Research Program of the Chinese Academy of Sciences under Grant No.XDA27030300.

Copyright © 2022 ISCA

10.21437/Interspeech.2022-914

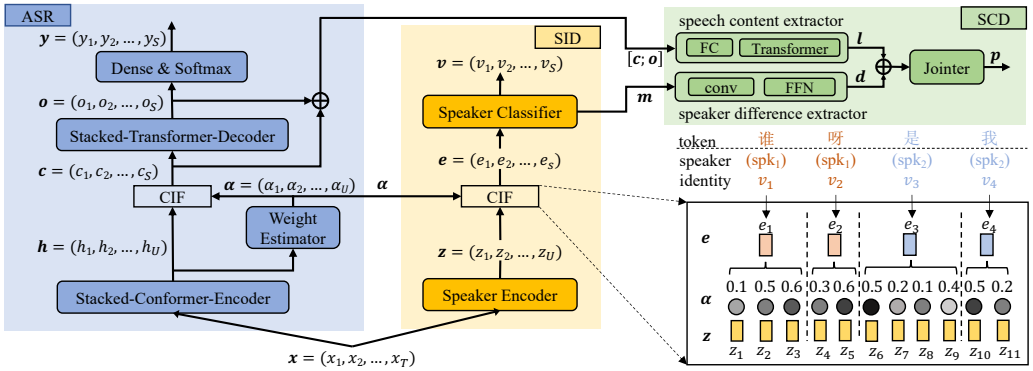

Figure 1: *Schematic diagram of the proposed model. The ASR and speaker encoder have extra pre-training. During the joint training of the ASR, SID and SCD, the parameters of ASR are frozen. The lower right corner of the figure uses an example to show how the CIF converts the frame-level speaker representations into token-level speaker representations. The dashed line in the box represents the token acoustic boundaries. The token-level speaker representations* $e_1 = 0.1*z_1+0.5*z_2+0.4*z_3$, $e_2 = 0.2*z_3+0.3*z_4+0.5*z_5$, $e_3 = 0.1*z_5+0.5*z_6+0.2*z_7+0.1*z_8+0.1*z_9$, $e_4 = 0.3*z_9+0.5*z_{10}+0.2*z_{11}$.

forms the acoustic feature sequence $\boldsymbol{x} = (x_1, x_2, ..., x_T)$ to a encoded output $\boldsymbol{h} = (h_1, h_2, ..., h_U)$, where T and U are the total length of the input sequence and the encoded output, respectively. A weight estimator calculates the information weights $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_U)$ for $\boldsymbol{h}$ at each time step. α_u scales the amount of information contained in h_u. In the training stage, a scaling operation is applied to the raw weights $\boldsymbol{\alpha}$ to ensure that the sum of the weights $\boldsymbol{\alpha}$ is equal to the length of targets $\bar{\boldsymbol{y}} = (\bar{y}_1, \bar{y}_2, ..., \bar{y}_S)$.

$$\boldsymbol{\alpha}' = \frac{S}{\sum_{u=1}^{U} \alpha_u} \cdot \boldsymbol{\alpha} \tag{1}$$

The CIF processes the frame-level speech representations $\boldsymbol{h}$ and information weights $\boldsymbol{\alpha}'$ in a left-to-right manner. The information weight α'_u is accumulated to α^a_u. Meanwhile, the token-level speech representations are integrated frame by frame.

$$\alpha^a_u = \alpha^a_{u-1} + \alpha'_u \tag{2}$$

$$h^a_u = h^a_{u-1} + \alpha'_u \cdot h_u \tag{3}$$

Once the accumulated weight α^a_u reaches a threshold β, a token-level acoustic boundary is located, and these time steps are called fired time steps, denoted as u^*. At each fired time step, the information weight α_{u^*} is divided into two parts, α_{u1} and α_{u2}. α_{u1} is used for completing the current integration of token-level representation c_i. The sum of information weights consumed in a complete integration is 1. α_{u2} remains for the next integration. A complete integration spans from the last fired time step to the current fired time step.

$$\alpha_{u1} = 1 - \alpha^a_{u^*-1} \tag{4}$$

$$\alpha_{u2} = \alpha'_{u^*} - \alpha_{u1} \tag{5}$$

$$c_i = h^a_{u^*-1} + \alpha_{u1} \cdot h_{u^*} \tag{6}$$

After firing the current integration results c_i (Eq. 6), the accumulated weight and token-level representation state will be reset as follows:

$$\alpha^a_{u^*} = \alpha_{u2} \tag{7}$$

$$h^a_{u^*} = \alpha_{u2} \cdot h_{u^*} \tag{8}$$

The process described from Eq. 2 to Eq. 8 will be repeated until the last time step U. Then the CIF finishes transferring the encoded output $\boldsymbol{h}$ into token-level representations $\boldsymbol{c} = (c_1, c_2, ..., c_S)$, and these fired time steps are located as token acoustic boundaries. Since we apply the scaling operation in Eq. 1 and the threshold β is set to 1, the length of $\boldsymbol{c}$ is equal to the length of target labels $\bar{\boldsymbol{y}}$. For the convenience of the following description, we summarize the process of integration and fire formulated from Eq. 1 to Eq. 8 as follow:

$$\boldsymbol{c} = \text{CIF}(\boldsymbol{h}, \boldsymbol{\alpha}) \tag{9}$$

The decoder receives the fired token-level representations $\boldsymbol{c}$ and predicts the label sequence $\bar{\boldsymbol{y}} = (\bar{y}_1, \bar{y}_2, ..., \bar{y}_S)$ in an autoregressive manner.

$$\Pr(y_i | \bar{y}_{1:i-1}, c_{1:i}) = \text{ASRDecoder}(\bar{y}_{i-1}, c_i) \tag{10}$$

During inference, the scaling operation in Eq. 1 is skipped. To alleviate this mismatch between training and inference, the quantity loss and tail handling are adapted. For more details of the CIF-based ASR, we refer readers to [21].

3. Proposed method

3.1. Overview

In this paper, we propose to use speaker difference and speech content to conduct a token-level speaker change detection (SCD). The model body is a joint framework for automatic speech recognition (ASR), speaker identification (SID) and SCD. The three parts are distinguished by different colors in Fig. 1. The ASR part transfers the input feature sequence $\boldsymbol{x}$ to the target token sequence $\boldsymbol{y}$ while providing the token acoustic boundaries contained in the information weights $\boldsymbol{\alpha}$ and speech representations $\boldsymbol{c}$ and $\boldsymbol{o}$. The SID part conducts a token-level speaker classification, and then generates token-level speaker embeddings $\boldsymbol{m}$. The SCD part captures the speaker difference $\boldsymbol{d}$ between the speaker embeddings $\boldsymbol{m}$, and combines it with the speech content $\boldsymbol{l}$ to detect speaker changes between tokens.

3.2. Model Structure

The details of the model body is shown in Fig.1. Given the input feature sequence $\boldsymbol{x} = (x_1, x_2, ..., x_T)$, the CIF-based ASR predicts the token sequence as described in Section 2.

The SID consists of a speaker encoder, a classifier, and the CIF module. The speaker encoder transfers $\boldsymbol{x}$ to frame-level speaker representations $\boldsymbol{z} = (z_1, z_2, ..., z_U)$, and it has the same temporal downsampling as the ASR encoder to make the length of their output sequence consistent. Then the CIF mechanism uses the frame-level speaker representations $\boldsymbol{z}$ and the information weights $\boldsymbol{\alpha}$ provided by the ASR part to generate token-level speaker representations $\boldsymbol{e}$.

$$\boldsymbol{e} = (e_1, e_2, ..., e_S) = \text{CIF}(\boldsymbol{z}, \boldsymbol{\alpha}) \tag{11}$$

In the calculation process of the CIF, the fired time steps and

the length of output sequence S are dependent on the information weights $\boldsymbol{\alpha}$, so the token-level speaker representations and speech representations can be one-by-one correspondence. The speaker classifier stacks two full-connected (FC) layers. The first layer transfers $\boldsymbol{e}$ to the token-level speaker embeddings $\boldsymbol{m} = (m_1, m_2, ..., m_S)$. The second layer projects $\boldsymbol{m}$ to the speaker probability distribution $\boldsymbol{v} = (v_1, v_2, ..., v_S)$.

The SCD part consists of a speech content extractor (SCE), a speaker difference extractor (SDE), and a jointer. The SCE stacks a FC layer and uni-directional transformer layers with masked multi-head self-attention [22]. It pre-processes the speech presentations from the CIF output $\boldsymbol{c}$ and ASR decoder output $\boldsymbol{o}$, and generates the speech content $\boldsymbol{l} = (l_1, l_2, ..., l_S)$. The SDE stacks a 1d-convolution layer and a feed-forward network (FFN) layer to calculate the speaker difference $\boldsymbol{d} = (d_1, d_2, ..., d_S)$ between the token-level speaker embeddings $\boldsymbol{m}$. Using 1D convolution with kernel size 3, the SDE calculates the difference between the speaker embeddings of the last token, the current token and the next token. Then the speaker difference and the speech content are concatenated and fed into the jointer. The FC layer based jointer combines these two cues to predict speaker change probability $\boldsymbol{p} = (p_1, p_2, ..., p_S)$.

3.3. Training

Before the joint training of the proposed model, we pre-train the CIF-based ASR part and speaker encoder, respectively. The loss used in the pre-training of ASR is consistent with our previous work [21]. The pre-training of the speaker encoder is an utterance-level speaker classification. We add an extra average pooling and an output FC layer upon the speaker encoder to perform the pre-training. The AMSoftmax [23] loss is used to capture more discriminative speaker embeddings.

During the joint training stage, the parameters of ASR part are frozen. The loss is the interpolation of a token-level AM-Softmax loss for speaker classification and a token-level binary cross-entropy (BCE) loss for speaker changes detection.

$$\mathcal{L}_{\text{joint}} = \text{AMS}(\boldsymbol{v}, \bar{\boldsymbol{v}}) + \text{BCE}(\boldsymbol{p}, \bar{\boldsymbol{p}}) \tag{12}$$

where $\bar{\boldsymbol{v}}$ and $\bar{\boldsymbol{p}}$ are token-level speaker identity label and speaker change label, respectively.

4. Experiments

4.1. Dataset

AISHELL-4 [24] is a public real-recorded conversation speech dataset in conference scenario. It contains 118 hours of audio recorded by an 8-channel microphone array. All our experiments only use the first channel. Each of the total 211 meeting sessions contains 4 to 8 speakers. As the speech recognition for the overlapped speech is beyond the scope of this paper, we exclude the overlapped speech of the AISHELL-4. The excluded rules are that (1) one to four consecutive intervals annotated in the original annotation file form a sentence. (2) these sentences containing overlapped speech for more than 1s will be discarded. (3) these sentences containing overlapped speech that accounts for more than 10% of the duration of any interval will be discarded. (4) these sentences containing more than 10s silence will be discarded. The training and test set are processed independently. Then we extract five meeting sessions as the development set from the original training set, and the remainder is used as our training set. The test set is further excluded these sentences containing no speaker change. The details of the processed data are shown in Table 1. In order to facilitate readers to reproduce our work, we release the data processing scripts[1].

Table 1: *Details of the processed data.*

	Train	**Dev**	**Test**
#Samples	84705	3586	1971
#Speakers	36	19	25
Min duration (s)	0.11	0.19	0.61
Mean duration (s)	7.90	7.68	11.75
Max duration (s)	71.87	46.28	60.78
Total duration (h)	185.79	7.65	6.43

4.2. Evaluating Metric

All our experiments are evaluated on the equal coverage-purity (ECP). The formula of the purity and the coverage [25] are as follows:

$$\text{Purity} = \frac{\sum_{h \in H} \max_{r \in R} |h \cap r|}{\sum_{h \in H} |h|} \tag{13}$$

$$\text{Coverage} = \frac{\sum_{r \in R} \max_{h \in H} |r \cap h|}{\sum_{r \in R} |r|} \tag{14}$$

where R and H are the set of segments cut by the reference and hypothesized change points, respectively. These points exceed a threshold θ are detected as speaker changes. $|s|$ is the duration of segment s. $r \cap h$ is the intersection of segments r and h.

4.3. Experimental Setup

We extract input features using the same setup as [21]. A convolutional layer with $1/2$ temporal downsampling is used as the front-end of the ASR, and the filter number is set to 64. The ASR encoder consists of 15 Conformer [26] encoder layers with 8 attention heads, 400 attention dimensions and 1600 FFN dimensions. There is a max-pooling layer for $1/2$ temporal downsampling after the fifth and tenth layers, respectively. The weight estimator consists of a 1-dimensional convolutional layer and a FC layer. The kernel size of the convolutional layer is set to 3, and the filter number is 400. The FC layer has one output unit with sigmoid activation. The ASR decoder consists of 2 Transformer [21] layers with 8 attention heads, 400 attention dimensions and 1600 FFN dimensions. The speaker encoder has the same architecture as the 18-layer ResNet in [27] except for not the final average pooling layer and halving the channels of all the convolutional layers. Both the speaker encoder and the ASR encoder have a $1/8$ temporal downsampling. The speaker decoder stacks two FC layers. The first layer has 128 units with ReLU activation, the second layer has 36 (total speaker number in the training set) output units with softmax nonlinearity. The speech content extractor consists of an FC layer and transformer layers. The FC layer projects the inputs to 400 dimensions. The structure of transformer layers is consistent with the ASR decoder. The speaker difference extractor consists of a 1d-convolutional layer with kernel size 3 and an FFN layer with 512 hidden dimensions. The jointer consists of a hidden FC layer with 528 units and an output FC layer with 1 unit. The threshold β in these two CIF is set to 1.

For the pre-training of ASR, the learning rate warms up for the first 1k iterations to a peak of 10^{-3} and holds on for the next 40k iterations, and then linearly decays for the remainder. For the pre-training of the speaker encoder, the SGD optimizer is used. The learning rate warms up for the first 1k iterations to a peak of 10^{-4} and holds on for the next 5k iterations. During

[1] https://github.com/zhiyunfan/aishell4-preprocess

the joint training stage, the learning rate warms up for the first 1k iterations to a peak of 10^{-3}, and holds on for the remainder.

4.4. Results

Table 2: *ECP/% for baseline systems and the proposed method.*

Exp	Model			ECP
A1	BSL with Bi-LSTM [28]			84.05
A2	BSL with ResNet18			85.77
	Speaker	**Difference**	**Speech content**	
B1	$\boldsymbol{z}$	×	×	82.08
B2	$\boldsymbol{z}$	✓	×	82.18
B3	$\boldsymbol{m}$	×	×	81.47
B4	$\boldsymbol{m}$	✓	×	86.99
B5	×	×	✓	86.73
B6	$\boldsymbol{m}$	✓	✓	**88.22**
	Convolution context			
C1	$[-1, 1]$			88.22
C2	$[-2, 2]$			87.61
C3	$[-3, 3]$			87.99

4.4.1. Baseline results

We build two baseline systems with the Pyannote1.1 toolkit [28]. The baseline system A1 with Bi-LSTM follows the structure in [10], which addresses the SCD as a binary sequence labeling (BSL) task. The input MFCC feature sequence is encoded by the Bi-LSTM. Then a multi-layer perceptron (MLP) projects the encoded sequence to a frame-level score sequence between 0 and 1. For a fair comparison, the baseline A2 replaces the Bi-LSTM with the ResNet18, the structure used in our speaker encoder. The results of these two baseline systems are shown in the A1 and A2 of Table 2.

4.4.2. Evaluation with token-level speaker difference

The B1 to B4 in Table 2 are four variants of the proposed model. All the four experiments ablate the use of the speech content. B1 feeds the frame-level speaker representations $\boldsymbol{z}$ into the SDE, and skips the convolutional layer in the SDE. Compared with B1, B2 uses the convolutional layer in the SDE to capture the speaker difference. In contrast, B3 and B4 feed the token-level speaker embeddings $\boldsymbol{m}$ into the SDE, and then the jointer detects speaker changes on the acoustic boundaries.

Comparing B2 with B1, we find that using convolution in the SDE to capture the speaker difference between the frame-level speaker representations $\boldsymbol{z}$ brings little improvement. Comparing B4 with B3, both of which use the token-level speaker embeddings $\boldsymbol{m}$, the use of speaker difference brings a 5.52% ECP improvement. These results indicate that calculating the speaker difference between token-level speaker embeddings is more useful to the SCD task. Moreover, compared with the frame-by-frame speaker change detection, detecting speaker changes on the token acoustic boundaries can greatly reduce the number of candidates to be processed. From the results of B1 to B4, we could find that the token-level speaker difference information (B4) obtains the best performance. Thus, all subsequent experiments use token-level speaker difference for the token-level SCD.

4.4.3. SCD with the speaker difference and the speech content

Another advantage of using token-level speaker difference is the ability to combine it with token-level speech content at the same granularity. B6 in Table 2 shows the result of the proposed method with both token-level speaker difference and speech content. The results of B4 and B5 represent that our method ablates speech content or speaker difference, respectively. We find that using one of the speaker difference or speech content in our method has outperformed these two baseline systems (A1 and A2), and B6 shows that combining these two cues can bring further performance improvement, and outperforms the baseline system with ResNet18 by 2.45% ECP.

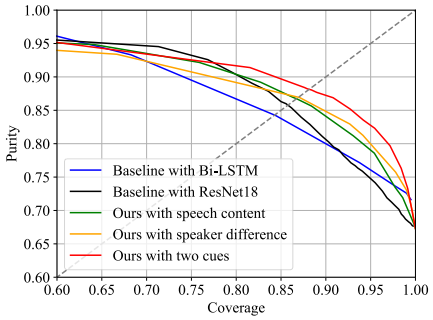

Figure 2: *Coverage-purity measured on baseline methods and our method with speaker difference and speech content.*

In Fig 2, we plot the coverage-purity curves of B4, B5, B6 and these two baseline systems (A1 and A2). All curves are obtained by varying the threshold θ. The dashed line is diagonal, where purity is equal to coverage. Among the five systems, area under the curve of B6 is the largest, and B6 performs better in most of the varing thresholds.

4.4.4. Evaluation with different speaker context settings

In the previous experiments, all convolutional layers in the SDE have the context of $[-1, 1]$, which means that the model refers to the speaker embedding of the previous token and the next token when determining whether there is a speaker change between current token and next token. We then attempt to improve the proposed method by more speaker contexts. The results of our method with different speaker contexts are shown from C1 to C3 in Table 2. The best ECP performance is achieved by the $[-1, 1]$ context. After our analysis of the corpus, we find a lot of speaker change points where one person says only one token inside another person's speech. For such cases, $[-1, 1]$ context is enough for the SCD, and more contexts bring confusion.

5. Conclusions

In this paper, we propose to use both speaker difference and speech content to detect speaker changes on the located token acoustic boundaries by the CIF. The experiments demonstrate the advantages of using the speaker difference and the speech content for the SCD task, and show performance improvement of detecting speaker changes on the token acoustic boundaries. Specifically, our token-level SCD system using both speaker difference and speech content outperforms the competitive frame-level baseline by 2.45% ECP. In the future, using the CIF to integrate the other information contained in speech with content for more speech processing tasks will be a potential research direction.

6. Acknowledgments

The authors wish to thank Yangcheng Wu for his assistance with the model training when he was an intern in Bytedance AI Lab.

7. References

[1] S. Chen, P. Gopalakrishnan *et al.*, "Speaker, environment and channel change detection and clustering via the bayesian information criterion," in *Proc. DARPA broadcast news transcription and understanding workshop*, 1998, pp. 127–132.

[2] M. A. Siegler, U. Jain, B. Raj, and R. M. Stern, "Automatic segmentation, classification and clustering of broadcast news audio," in *Proc. DARPA speech recognition workshop*, 1997, pp. 97–99.

[3] V.-A. Tran, V. Le, C. Barras, and L. Lamel, "Comparing multistage approaches for cross-show speaker diarization," in *International Speech Communication Association (ISCA)*, 2011, pp. 1053–1056.

[4] M. Cettolo, M. Vescovi, and R. Rizzi, "Evaluation of bic-based algorithms for audio segmentation," *Computer Speech & Language*, pp. 147–170, 2005.

[5] H. Gish, M.-H. Siu, and J. R. Rohlicek, "Segregation of speakers for speech recognition and speaker identification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1991, pp. 873–876.

[6] Z. Ge, A. N. Iyer, S. Cheluvaraja, and A. Ganapathiraju, "Speaker change detection using features through a neural network speaker classifier," in *Intelligent Systems Conference (IntelliSys)*, 2017, pp. 1111–1116.

[7] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A review of recent research," *Transactions on Audio, Speech, and Language Processing*, pp. 356–370, 2012.

[8] A. S. Malegaonkar, A. M. Ariyaeeinia, and P. Sivakumaran, "Efficient speaker change detection using adapted gaussian mixture models," *Transactions on Audio, Speech, and Language processing*, pp. 1859–1869, 2007.

[9] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair, "Stream-based speaker segmentation using speaker factors and eigenvoices," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 4133–4136.

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

[11] L. Sarı, S. Thomas, M. Hasegawa-Johnson, and M. Picheny, "Pre-training of speaker embeddings for low-latency speaker change detection in broadcast news," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6286–6290.

[12] L. Sari, M. Hasegawa-Johnson, and S. Thomas, "Auxiliary networks for joint speaker adaptation and speaker change detection," *Transactions on Audio, Speech, and Language Processing*, pp. 324–333, 2020.

[13] M. Hrúz and Z. Zajíc, "Convolutional neural network for speaker change detection in telephone speaker diarization system," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 4945–4949.

[14] A. O. Hogg, C. Evers, and P. A. Naylor, "Speaker change detection using fundamental frequency with application to multi-talker segmentation," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5826–5830.

[15] T. Kemp, M. Schmidt, M. Westphal, and A. Waibel, "Strategies for automatic segmentation of audio data," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2000, pp. 1423–1426.

[16] R. Li, T. Schultz, and Q. Jin, "Improving speaker segmentation via speaker identification and text segmentation," in *Interspeech*, 2009, pp. 904–907.

[17] Z. Meng, L. Mou, and Z. Jin, "Hierarchical rnn with static sentence-level attention for text-based speaker change detection," in *Conference on Information and Knowledge Management (CIKM)*, 2017, pp. 2203–2206.

[18] T. J. Park and P. Georgiou, "Multimodal speaker segmentation and diarization using lexical and acoustic cues via sequence to sequence neural networks," *arXiv preprint arXiv:1805.10731*, 2018.

[19] M. À. India Massana, J. A. Rodríguez Fonollosa, and F. J. Hernando Pericás, "Lstm neural network-based speaker segmentation using acoustic and language modelling," in *International Speech Communication Association (ISCA)*, 2017, pp. 2834–2838.

[20] O. H. Anidjar, I. Lapidot, C. Hajaj, A. Dvir, and I. Gilad, "Hybrid speech and text analysis methods for speaker change detection," *Transactions on Audio, Speech, and Language Processing*, pp. 2324–2338, 2021.

[21] L. Dong and B. Xu, "Cif: Continuous integrate-and-fire for end-to-end speech recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6079–6083.

[22] A. Nicolson and K. K. Paliwal, "Masked multi-head self-attention for causal speech enhancement," *Speech Communication*, pp. 80–96, 2020.

[23] Y. Liu, L. He, and J. Liu, "Large margin softmax loss for speaker verification," in *International Speech Communication Association (ISCA)*, 2019, pp. 2873–2877.

[24] Y. Fu, L. Cheng, S. Lv, Y. Jv, Y. Kong, Z. Chen, Y. Hu, L. Xie, J. Wu, H. Bu *et al.*, "Aishell-4: An open source dataset for speech enhancement, separation, recognition and speaker diarization in conference scenario," in *International Speech Communication Association (ISCA)*, 2021, pp. 3665–3669.

[25] H. Bredin, "pyannote.metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems," in *International Speech Communication Association (ISCA)*, 2017, pp. 3587–3591.

[26] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," in *International Speech Communication Association (ISCA)*, 2020, pp. 5036–5040.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[28] H. Bredin, R. Yin, J. M. Coria, G. Gelly, P. Korshunov, M. Lavechin, D. Fustes, H. Titeux, W. Bouaziz, and M.-P. Gill, "Pyannote.audio: neural building blocks for speaker diarization," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7124–7128.