AUGMENTING TRANSFORMER-TRANSDUCER BASED SPEAKER CHANGE DETECTION WITH TOKEN-LEVEL TRAINING LOSS

Guanlong Zhao, Quan Wang, Han Lu, Yiling Huang, Ignacio Lopez Moreno

Google LLC, USA

{guanlongzhao,quanw,luha,yilinghuang,elnota}@google.com

ABSTRACT

In this work we propose a novel token-based training strategy that improves Transformer-Transducer (T-T) based speaker change detection (SCD) performance. The conventional T-T based SCD model loss optimizes all output tokens equally. Due to the sparsity of the speaker changes in the training data, the conventional T-T based SCD model loss leads to sub-optimal detection accuracy. To mitigate this issue, we use a customized edit-distance algorithm to estimate the token-level SCD false accept (FA) and false reject (FR) rates during training and optimize model parameters to minimize a weighted combination of the FA and FR, focusing the model on accurately predicting speaker changes. We also propose a set of evaluation metrics that align better with commercial use cases. Experiments on a group of challenging real-world datasets show that the proposed training method can significantly improve the overall performance of the SCD model with the same number of parameters.

***Index Terms*—** Speaker change detection, speaker turn detection, Transformer-Transducer, minimum Bayes risk training

1. INTRODUCTION

Speaker change detection (SCD) or speaker turn detection [1–4] is the process of identifying the speaker change points in a multi-speaker continuous conversational input audio stream. SCD has broad applications in enhancing speaker diarization accuracy [4, 5], improving automatic speech recognition quality [6], and generating line breaks in captions to augment readability and accessibility [7].

Conventionally, SCD is achieved by using a neural network to map acoustic features (e.g., spectral/cepstral coefficients of various flavors [3, 8, 9]) or biometric features [10] (e.g., speaker embeddings [11–14]) to a frame or segment level binary prediction — yes/no speaker change. The neural network is generally trained by minimizing the binary cross entropy loss between the ground-truth speaker change labels and the predictions. These conventional approaches have various limitations. First, they require accurate timing information of the speaker change point, which is difficult since marking speaker change points is a highly subjective process for human annotators. Second, the methods that use purely acoustic information ignore rich semantic information in the audio signal. For example, by only looking at the text transcript of the conversation "How are you I'm good", we can confidently conjecture there is a speaker change between "How are you" and "I'm good". Third, the methods that use speaker embeddings for SCD utilize sensitive biometric information that can be exploited for unintended purposes. Thus, they are sub-optimal from a privacy point of view [15].

To mitigate the aforementioned issues, previously we have proposed an SCD model using a Transformer-Transducer (T-T) [4]. Specifically, we augment the text transcription of the spoken utterance with a special speaker turn token <st>, and then train the model to output both regular text tokens and the special speaker turn token. This model does not need accurate timestamps for training since the T-T model is trained in a seq2seq fashion and does not need forced-alignment to provide training targets. The model also utilizes both acoustic and linguistic information in the input audio.

Speaker turns are relatively sparse compared to regular spoken words. Based on estimates on the training data we use, regular spoken words appear 40+ times more frequently than speaker turns. The conventional T-T training loss maximizes the log probability of the entire output sequence, including both spoken words and the special speaker turn token. The sparsity of speaker changes in the training data leads to them being de-emphasized, resulting in high error rates in the inference. In the ASR research community, minimum Bayes risk (MBR) training [16] is a widely used technique to improve model performance. One common variation of this technique is the Edit-distance based MBR (EMBR)[1] technique, which optimizes model parameters to minimize the expected word error rate [17]. Inspired by the EMBR technique, we propose to supplement the original T-T training loss with additional token-based penalty terms to minimize the expected recognition error of the speaker turns. During training, we apply a constrained edit distance algorithm to identify the speaker change false acceptance (FA) and false rejection (FR) errors in the N-best hypotheses, and then penalize the training loss to steer the gradient away from the high FA/FR region.

The contributions of this work are two-fold. First, we propose a training loss for SCD that directly minimizes the expected token-level FA and FR rates, and results in improved SCD performance. Second, we define a new set of SCD evaluation metrics and demonstrate that they can better reflect the model quality than previous metrics [18] on a group of diverse test sets.

2. SYSTEM DESCRIPTION

2.1. Baseline SCD model

The recurrent neural network transducer (RNN-T) [19] is an ASR model architecture that can be trained end-to-end with the RNN-T loss. An RNN-T model includes an audio encoder, a label encoder, and a joint network that produces the output softmax distribution over a predefined vocabulary. We adopt the Transformer Transducer [20], a variant of the RNN-T architecture, as the speaker change detection model for its advantages of faster inference speed and better handling of long-form deletion issues. We use LSTM layers as the label encoder, and fully connected layers as the joint network.

To create training targets, we add a special speaker turn token <st> between two different speakers' transcripts (e.g. "hello how are you <st> I am good <st>") to model speaker turns during training. This is inspired by [21] that adds speaker roles as part of the transcript (e.g. "hello how are you <spk:dr> I am good <spk:pt>"). Compared with audio-only SCD models [3, 9], this

[1] Also known as Minimum Word Error Rate (MWER).

model can potentially utilize the language semantics as a signal for speaker segmentation. T-T is trained in a seq2seq fashion, where the input sequence contains log-Mel filterbank energy features, and the output sequence contains the transcript that includes both transcript texts and the special speaker turn tokens. For inference, we perform a beam-search with the T-T SCD model's softmax outputs, and identify the speaker turn tokens. We use the timestamps of the predicted speaker turn tokens in the evaluation.

2.2. Token-based training loss

To focus modeling capacity on the SCD task, we augment the training loss with an additional token-based SCD penalty. On a high level, we first construct a T-T model that takes audio as input, and outputs the speaker turn augmented transcriptions. We optimize the T-T model by maximizing the log probability and following the process described in Sec. 2.1. We then warm-start a new T-T model with the model trained in the previous step and fine-tune the new model on the same training data with the following steps: (1) for each training utterance, perform a beam search to get the N-best hypotheses associated with their corresponding probability scores, which is how likely the hypotheses appear based on the existing model parameters; (2) compute the token-level FA and FR from the N-best hypotheses; (3) fine-tune the model with a loss function that is a weighted sum of the log probability and token-level FA and FR rates.

2.2.1. Token-level FA and FR

Mathematically, let M be the number of training samples and N be the number of hypotheses per training sample; let $\mathbf{H}_{ij}$ be the j-th hypothesis of the i-th training sample, where $i \in [1, M]$ and $j \in [1, N]$; let $\mathbf{P}_{ij}$ be the probability score associated with $\mathbf{H}_{ij}$ given by the model; let $\mathbf{R}_{ij}$ be the corresponding reference transcription. We first compute a customized minimum edit distance alignment [22] between all $\mathbf{H}_{ij}$ and $\mathbf{R}_{ij}$. The idea of the customized minimum edit distance alignment is to only allow substitutions among regular spoken words, and each special speaker turn token prediction `<st>` can only be one of {`correct`, `deleted`, `inserted`}. A substitution error between a regular spoken word token and the special speaker turn token is ill-defined, thus, not allowed in the customized minimum edit distance alignment. To achieve this, the edit distance algorithm applies the following costs for its optimization,

$$\text{sub-cost}(r, h) = \begin{cases} 0, & \text{If } r = h; \\ 1, & \text{If } r \neq h \neq \texttt{<st>}; \\ +\infty, & \text{Otherwise.} \end{cases} \tag{1}$$

$$\text{ins/del-cost}(token) = \begin{cases} k, & \text{If } token = \texttt{<st>}; \\ 1, & \text{Otherwise.} \end{cases} \tag{2}$$

Here, r and h are tokens in $\mathbf{R}_{ij}$ and $\mathbf{H}_{ij}$, respectivly. The constant $k \geq 1$ controls the tolerance of the offset in predicting `<st>`. If $k = 1$, we expect an exact match between the reference and predicted `<st>` tokens. If $k > 1$, we allow a maximum offset of $\lfloor k \rfloor$ tokens between a pair of reference and predicted `<st>` tokens for them to be considered as correctly aligned, offering some tolerance on annotation errors.

2.2.2. Training loss

Based on the optimal alignment obtained from the customized minimum edit distance, we identify the number of speaker turn token insertions (denoted as $\mathbf{FA}_{ij}$) and deletions ($\mathbf{FR}_{ij}$) as well as the number of spoken word errors $\mathbf{W}_{ij}$ in $\mathbf{H}_{ij}$. We compute the per sample token-level loss as

$$\mathbf{L}_{ij} = \mathbf{P}_{ij} \cdot \frac{\alpha \mathbf{W}_{ij} + \beta \mathbf{FA}_{ij} + \gamma \mathbf{FR}_{ij}}{\mathbf{Q}_{ij}}, \tag{3}$$

where $\{\alpha, \beta, \gamma\}$ control the relative strength of each subcomponent, and $\mathbf{Q}_{ij}$ is the total number of tokens in $\mathbf{R}_{ij}$. We generally set β and γ to be much larger than α to force the model to reduce the speaker change insertion and deletion rates. In essence, Eq. (3) is a weighted sum of the expected value of WER, SCD FA, and SCD FR. We compute the final per batch training loss as

$$L_{\text{SCD}} = \sum_{i=1}^{M} \sum_{j=1}^{N} \mathbf{L}_{ij} - \lambda \log P(\mathbf{Y}|\mathbf{X}), \tag{4}$$

where $-\log P(\mathbf{Y}|\mathbf{X})$ is the negative log probability of the ground-truth transcription $\mathbf{Y}$ conditioned on the input acoustic features $\mathbf{X}$. The regularization term λ controls the strength of the negative log probability loss.

3. EVALUATION METRICS

Conventional evaluation metrics for the speaker change detection task are timestamp-based precision and recall rates [8]. A predicted speaker change point is considered as correct if it falls within a temporal window (the "collar") surrounding a reference speaker change point. Therefore, these metrics are very sensitive to the temporal precision of human annotations, and the precision and recall rates quickly reach zero when the collar approaches zero.

3.1. Interval-based precision and recall

In applications such as line breaking in captioning, we argue that speaker changes should be treated as time intervals instead of time-stamped change "points". For example, in a conversational recording, if speaker A spoke from 0.1-10.5s and speaker B spoke from 10.8-15.3s, the time interval of 10.5-10.8s is where the speaker change happened. If an SCD system predicts a speaker change within this speaker change interval, we should treat this prediction as correct. Following this argument, we propose a new way to compute SCD precision and recall rates by matching the predicted speaker changes to the ground-truth speaker change intervals. We assume the following when computing these metrics: (1) we use speaker annotations to infer speaker change intervals; (2) the test data have "dense" speaker annotations. "Dense" annotation means that the test utterance has speaker labels throughout and there are no large unannotated chunks of audio. As an example, it is considered bad data if the annotator labeled one minute of speech, then skipped 10 minutes, and annotated another two minutes of speech. The metrics are computed as follows (refer to Fig. 1 for the visualization of different components).

First, based on the reference speaker annotations, we derive the set of mono-speaker time ranges (denoted as U) as well as the min-start (T_{min}) and max-end (T_{max}) time of all speaker annotations. For a densely annotated recording, T_{min} and T_{max} should generally be the start and end times of the entire utterance. However, there may be cases where the annotations do not cover the full utterance, for example, to skip initial and trailing non-speech audio.

Within the $[T_{\text{min}}, T_{\text{max}}]$ time range, we take the complement set ($\overline{U}$) of the mono-speaker time ranges (U) and treat them as speaker change intervals. $\overline{U}$ includes multi-speaker segments and speaker switching segments. We drop the speaker change predictions that are outside of $[T_{\text{min}}, T_{\text{max}}]$ and do not count them during scoring because it is impossible to know whether the predictions outside of the min-max speaker annotation range are valid or not.

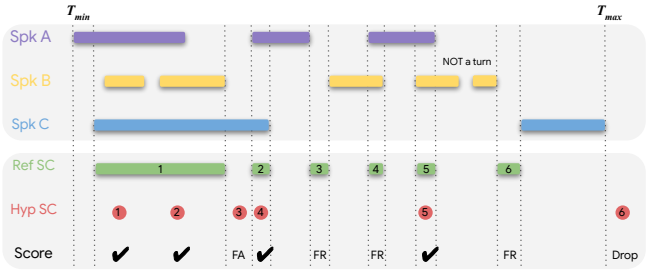

Fig. 1: Illustration of the various components for computing the precision and recall rates. "Spk A-C" stands for speaker annotations on a conversational utterance. "Ref SC" is the speaker change intervals (i.e., $\overline{U}$). "Hyp SC" is the predicted speaker change. "Score" shows the scoring decision of each prediction and reference.

For each speaker change prediction, we match it with the reference intervals in $\overline{U}$. If any of the reference intervals overlaps with the prediction, we mark the prediction as correct, otherwise the prediction is an FA. We compute the *precision* rate as the ratio between the number of correct predictions and the total number of predictions. For each reference speaker change interval, if any of the predictions can be matched with it (i.e., overlap in time, aka "hit"), we consider that reference speaker change interval as being correctly predicted, otherwise the reference speaker change interval is counted as an FR. We compute the *recall* rate as the ratio between the number of "hit" speaker change intervals and the total number of speaker change intervals. We note that one can also calculate the *recall* based on the duration of the intervals instead of counting the numbers, which would favor longer speaker change intervals.

When performing the matching between speaker change predictions and the reference intervals, we allow a collar of e.g., 250ms, which is common in computing speaker diarization metrics [23].

3.2. Purity and coverage

Purity and coverage scores [3] are commonly used evaluation metrics in SCD tasks. For each reference speaker segment, we find its most overlapping hypothesis speaker segment (defined by the speaker change predictions) and obtain their intersection. The coverage is computed as the ratio between the duration of the intersection and the reference segment. Purity is the dual metric of coverage by swapping the roles of the reference and hypothesis segments in the calculation. By definition, coverage and purity scores focus on the mono-speaker segments in the audio, and therefore can be viewed as a set of complementary metrics to the precision and recall metrics defined above, which focus on evaluating the SCD performance in the speaker change intervals (i.e., non mono-speaker segments).

4. EXPERIMENTAL SETUP

4.1. Datasets

We train the T-T model using the Fisher corpus [24], the training subset of Callhome American English [25], the training subset (Full-corpus-ASR partition) of the AMI corpus [26], the ICSI corpus [27] training subset (we used the data splits from Kaldi [28]), an internal long-form dataset containing around 7,500 hours of audio, and an internal vendor-provided dataset that contains 80 hours of simulated business meeting recordings constructed by having participants discuss randomly assigned business topics with action items. Training utterances are segmented into 15-second segments with speaker turn tokens added so they can fit into the memory of TPUs [29].

Details of our evaluation datasets are listed in Table 1. For the first DIHARD challenge evaluation subset (DIHARD1) [30], we removed all YouTube-derived utterances. For Fisher, we withheld a subset[2] of 172 utterances for testing (not used in training). "Outbound" and "Inbound" are vendor-provided call center telephone conversations between call center attendants and customers, initiated by the call center and by customers, respectively. "Outbound" and "Inbound" were previously used in [4, 5].

Table 1: The test sets. For each set, we show the average number of speaker turns per minute and the average per recording duration.

Testset	Domain	Dur. (h)	Average	
			Turns/min	Dur./Rec. (min)
AMI	Meeting	9.1	10	34
Callhome	Telephone	1.7	19	5
DIHARD1	Mixed	16.2	12	9
Fisher	Telephone	28.7	13	10
ICSI	Meeting	2.8	13	55
Inbound	Telephone	21.0	9	5
Outbound	Telephone	45.6	13	6

All internal datasets were collected according to Google's Privacy Principles [31] and abide by Google AI Principles [32].

4.2. System configurations

We extract 128-dim log-Mel filter-bank energies from a 32ms window, stack every 4 frames, and sub-sample every 3 frames, to produce a 512-dimensional acoustic feature vector with a stride of 30 ms as the input to the acoustic encoder. The audio encoder shares the same model architecture and hyper-parameters as in [4] except that in this work we change the "Dense layer 2" to 512 dimensions to increase model capacity. For the label encoder, we use a single 128-dim LSTM layer. The text tokens are projected into a 64-dim embedding vector before feeding into the label encoder. For the joint network, we have a projection layer that projects both the audio and label encoder outputs to 512-dim. At the output of the joint network, it produces a distribution over 75 possible graphemes (the English alphabet, punctuation, the speaker change token `<st>`, and special symbols like "$") with a softmax layer. The model has around 27M parameters in total. For optimization, we follow the same hyper-parameters as described in [20]. We evaluate three systems that share the same model architecture,

- **Baseline**: Trained with the negative log probability loss.
- **EMBR**: Warm-started from the model parameters of the **Baseline** model and fine-tuned with a linear combination of the EMBR loss L_{EMBR} (Eq. (8) in [17] computed on the 4-best hypotheses) and the negative log probability. The final loss is formulated as $L_{\text{EMBR}} - 0.03 \log P(\mathbf{Y}|\mathbf{X})$. This serves as another baseline.
- **SCD loss** (proposed): Warm-started from the parameters of the **Baseline** model and fine-tuned with the L_{SCD} loss in Eqs. (3) and (4), and computed on the 4-best hypotheses. Empirically, we set $\alpha, \beta, \gamma, \lambda$ to 1, 10, 10, 0.03, respectively. We set $k = 1.1$ in Eq. (2).

5. RESULTS AND DISCUSSION

Due to space constraints[3], here we only report the recall rates described in Sec. 3.1 based on interval counts. Note that recall rates based on the interval duration follow the same trend. We compute purity and coverage rates with the *pyannote.metrics* package [18].

[2] https://github.com/google/speaker-id/blob/master/publications/ScdLoss/eval/fisher.txt

[3] Supplemental results and resources can be found at https://github.com/google/speaker-id/blob/master/publications/ScdLoss

Table 2: Long-form evaluation results. The last column shows the evaluation metrics computed by pooling all test sets together.

Evaluation Metric	System	AMI	CallHome	DIHARD1	Fisher	ICSI	Inbound	Outbound	Pooled data
Precision (%)	Baseline	80.9	81.0	78.7	81.8	78.7	73.0	76.3	78.1
	EMBR	**81.3**	**82.0**	**79.8**	**83.5**	**79.3**	**74.3**	**77.0**	**79.1**
	SCD loss	79.4	**82.0**	78.8	82.6	77.8	72.8	75.1	77.6
Recall (%)	Baseline	64.0	50.6	49.2	62.4	54.3	62.2	50.9	55.8
	EMBR	64.2	53.4	49.5	71.1	53.6	71.8	53.6	60.3
	SCD loss	**68.1**	**59.1**	**52.4**	**75.7**	**58.7**	**79.2**	**58.7**	**65.2**
F1 (%) (Precision & Recall)	Baseline	71.5	62.3	60.6	70.8	64.2	67.2	61.1	65.1
	EMBR	71.7	64.7	61.1	76.8	64.0	73.0	63.2	68.5
	SCD loss	**73.3**	**68.7**	**62.9**	**79.0**	**66.9**	**75.9**	**65.9**	**70.9**
Purity (%)	Baseline	87.4	84.3	90.3	80.5	76.9	95.0	76.7	82.7
	EMBR	87.6	84.1	90.5	82.7	77.0	95.3	77.1	83.5
	SCD loss	**88.5**	**84.9**	**91.0**	**83.5**	**77.7**	**95.5**	**78.3**	**84.3**
Coverage (%)	Baseline	**70.0**	**85.6**	64.9	**80.8**	79.3	**77.1**	83.4	**78.5**
	EMBR	70.0	85.3	**65.1**	80.6	**79.8**	76.7	**83.7**	**78.5**
	SCD loss	68.7	84.7	64.7	80.2	78.9	75.0	82.4	77.5
F1 (%) (Purity & Coverage)	Baseline	**77.8**	**84.9**	75.6	80.6	78.1	**85.1**	79.9	80.5
	EMBR	**77.8**	84.7	**75.7**	81.6	**78.4**	85.0	**80.3**	**80.9**
	SCD loss	77.3	84.8	75.6	**81.9**	78.3	84.0	**80.3**	**80.8**

5.1. Long-form results

The evaluation results on the original long-form data are summarized in Table 2 and the metrics follow the definitions in Sec. 3. The F1 score is the harmonic average of the two metrics involved. Overall, the three systems perform similarly on the purity-and-coverage F1 score, with the **EMBR** and **SCD loss** systems performing slightly better than the **Baseline**. On the precision-and-recall F1 score, the proposed **SCD loss** system outperforms the **Baseline** and **EMBR** systems by 8.9% and 3.5% relative, respectively. More specifically, the **SCD loss** improves the recall rate by 16.8% relative compared with the **Baseline** system while maintaining a similar level of precision (only a -0.6% relative regression). The DIHARD1, Inhound, and Outbound test sets do not have corresponding training sets, therefore, the precision-and-recall F1 score improvements on these test sets demonstrate that the proposed training technique can generalize to out-of-domain data.

For comparison, we also calculate the precision-and-recall F1 scores following their conventional definitions [3] (not shown in Table 2) using the implementation in the *pyannote.metrics* package. We set the collar value to 250ms on each side (the same as what we used in our precision and recall calculation). Overall, the **SCD loss** system has an F1 score of 36.5, outperforming the **Baseline** (32.2) by 13.4% relative, and the **EMBR** system (34.6) by 5.5% relative. Although the relative quality difference across systems remain consistent with the metrics we define in this work, the absolute value of the *pyannote.metrics* precision and recall metrics are not representative of the true system performance (as reflected by the coverage and purity measures) due to the small collar value, as discussed in Sec. 3. Simply increasing the collar value (e.g, to >1s) would not solve this issue since that would make these metrics too lenient on errors.

Table 3: Short-form results by pooling all test sets together.

Length	System	F1 (Precision & Recall)	F1 (Purity & Coverage)
30s	Baseline	55.2	75.9
	EMBR	60.9	80.8
	SCD loss	**65.0**	**81.5**
60s	Baseline	58.6	77.9
	EMBR	64.4	**81.1**
	SCD loss	**67.9**	**81.1**
120s	Baseline	61.8	79.5
	EMBR	66.6	**81.2**
	SCD loss	**69.6**	81.0

5.2. Short-form results

We are also interested in the models' performance on short utterances since a wide range of applications focus on recordings that are less than a couple of minutes in duration. Therefore, we segment the long-form data into shorter utterances with various target lengths (30s, 60s, 120s). Note that the segmentation was done based on speaker annotations to avoid chopping in the middle of a sentence. Results are summarized in Table 3. On the precision-and-recall F1 score, the **SCD loss** system performs the best across all conditions. On the purity-and-coverage F1 score, the **SCD loss** and **EMBR** systems perform similarly while outperforming the **Baseline** system. The **SCD loss** system performs better when the recordings are around 30s long, and the **EMBR** system performs slightly better when the segments are longer (120s). One possible explanation is that the training data is around 15s on average, so the **SCD loss** model is tuned towards performing better on shorter segments.

6. CONCLUSIONS

We introduce a novel token-based training loss that directly minimizes the SCD error rates for a Transformer-Transducer based SCD model. We also propose a new set of definitions for calculating the precision and recall rates for SCD evaluation. Experiments on a set of diverse evaluation sets demonstrate that the proposed training loss can significantly improve the recall rate of SCD while maintaining the precision rate. We show that the proposed new metrics can highlight model quality differences when the conventional purity and coverage scores cannot, hence providing additional insights for model improvements.

7. ACKNOWLEDGEMENTS

The authors would like to thank Jason Pelecanos, Hank Liao, Olivier Siohan, Françoise Beaufays, Pedro Moreno Mengibar, and Haşim Sak for their help.

8. REFERENCES

[1] Scott Chen and P.S. Gopalakrishnan, "Speaker, environment and channel change detection and clustering via the bayesian information criterion," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, 1998, vol. 8, pp. 127–132.

[2] Jitendra Ajmera, Iain McCowan, and Hervé Bourlard, "Robust speaker change detection," *IEEE Signal Processing Letters*, vol. 11, no. 8, pp. 649–651, 2004.

[3] Ruiqing Yin, Hervé Bredin, and Claude Barras, "Speaker change detection in broadcast TV using bidirectional long short-term memory networks," in *Proc. Interspeech*, 2017, pp. 3827–3831.

[4] Wei Xia, Han Lu, Quan Wang, Anshuman Tripathi, Yiling Huang, Ignacio Lopez Moreno, and Hasim Sak, "Turn-to-diarize: Online speaker diarization constrained by transformer transducer speaker turn detection," in *Proc. ICASSP*, 2022, pp. 8077–8081.

[5] Quan Wang, Yiling Huang, Han Lu, Guanlong Zhao, and Ignacio Lopez Moreno, "Highly efficient real-time streaming and fully on-device speaker diarization with multi-stage clustering," *arXiv:2210.13690*, 2022.

[6] Leda Sari, Mark Hasegawa-Johnson, and Samuel Thomas, "Auxiliary networks for joint speaker adaptation and speaker change detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 324–333, 2020.

[7] Gregor Donabauer, Udo Kruschwitz, and David Corney, "Making sense of subtitles: Sentence boundary detection and speaker change detection in unpunctuated texts," in *Companion Proceedings of the Web Conference 2021*, 2021, pp. 357–362.

[8] Marek Hrúz and Zbyněk Zajíc, "Convolutional neural network for speaker change detection in telephone speaker diarization system," in *Proc. ICASSP*, 2017, pp. 4945–4949.

[9] Ruiqing Yin, Hervé Bredin, and Claude Barras, "Neural speech turn segmentation and affinity propagation for speaker diarization," in *Proc. Interspeech*, 2018, pp. 1393–1397.

[10] Hagai Aronowitz and Weizhong Zhu, "Context and uncertainty modeling for online speaker change detection," in *Proc. ICASSP*, 2020, pp. 8379–8383.

[11] Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[12] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. ICASSP*, 2018, pp. 5329–5333.

[13] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proc. ICASSP*, 2014, pp. 4052–4056.

[14] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno, "Generalized end-to-end loss for speaker verification," in *ICASSP*. IEEE, 2018, pp. 4879–4883.

[15] Sam De Silva and Anthony Liu, "Europe's tough new law on biometrics," *Biometric Technology Today*, vol. 2017, no. 2, pp. 5–7, 2017.

[16] Karel Veselỳ, Arnab Ghoshal, Lukás Burget, and Daniel Povey, "Sequence-discriminative training of deep neural networks," in *Proc. Interspeech*, 2013, pp. 2345–2349.

[17] Matt Shannon, "Optimizing expected word error rate via sampling for speech recognition," in *Proc. Interspeech*, 2017, pp. 3537–3541.

[18] Hervé Bredin, "pyannote.metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems," in *Proc. Interspeech*, 2017, pp. 3587–3591.

[19] Alex Graves, "Sequence transduction with recurrent neural networks," *arXiv:1211.3711*, 2012.

[20] Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *Proc. ICASSP*, 2020, pp. 7829–7833.

[21] Laurent El Shafey, Hagen Soltau, and Izhak Shafran, "Joint speech recognition and speaker diarization via sequence transduction," in *Proc. Interspeech*, 2019, pp. 396–400.

[22] Daniel Jurafsky and James H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, chapter 3, pp. 74–77, Pearson, 2nd edition, 2008.

[23] Jonathan G Fiscus, Jerome Ajot, and John S Garofolo, "The rich transcription 2007 meeting recognition evaluation," in *Multimodal Technologies for Perception of Humans*, pp. 373–389. Springer, 2007.

[24] Christopher Cieri, David Miller, and Kevin Walker, "The Fisher corpus: A resource for the next generations of speech-to-text.," in *LREC*, 2004, vol. 4, pp. 69–71.

[25] A Canavan, D Graff, and G Zipperlen, "CALLHOME American English speech LDC97S42," LDC Catalog. Philadelphia: Linguistic Data Consortium, 1997.

[26] Jean Carletta et al., "The AMI meeting corpus: A pre-announcement," in *Machine Learning for Multimodal Interaction*, 2006, pp. 28–39.

[27] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters, "The ICSI meeting corpus," in *Proc. ICASSP*, 2003, vol. 1, pp. 364–367.

[28] "Kaldi ICSI data split," https://github.com/kaldi-asr/kaldi/blob/master/egs/icsi/README.txt, Accessed: 2022-10-17.

[29] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al., "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12.

[30] Neville Ryant, Kenneth Church, Christopher Cieri, Alejandrina Cristia, Jun Du, Sriram Ganapathy, and Mark Liberman, "First DIHARD challenge evaluation plan," Tech. Rep., Linguistic Data Consortium, University of Pennsylvania, 2018.

[31] "Google's privacy principles," https://googleblog.blogspot.com/2010/01/googles-privacy-principles.html, Accessed: 2022-10-17.

[32] "Artificial intelligence at Google: Our principles," https://ai.google/principles, Accessed: 2022-10-17.