THE SPEED SUBMISSION TO DIHARD II: CONTRIBUTIONS & LESSONS LEARNED

Md Sahidullah Jose Patino Samuele Cornell
Ruiqing Yin Sunit Sivasankaran Herve Bredin
Pavel Korshunov Alessio Brutti Romain Serizel
Emmanuel Vincent Nicholas Evans
Sébastien Marcel Stefano Squartini
Claude Barras

Idiap-RR-14-2019

NOVEMBER 2019

Centre du Parc, Rue Marconi 19, P.O. Box 592, CH - 1920 Martigny
T +41 27 721 77 11 F +41 27 721 77 12 info@idiap.ch www.idiap.ch

The *Speed* Submission to DIHARD II: Contributions & Lessons Learned

M. Sahidullah[1⋆], *J. Patino*[2⋆], *S. Cornell*[3⋆], *R. Yin*[4⋆], *S. Sivasankaran*[1⋆], *H. Bredin*[4⋆], *P. Korshunov*[5⋆], *A. Brutti*[6⋆], *R. Serizel*[1], *E. Vincent*[1], *N. Evans*[2], *S. Marcel*[5], *S. Squartini*[3⋆], *C. Barras*[4]

[1]LORIA, CNRS, Inria, Nancy, France, [2]EURECOM, Sophia Antipolis, France
[3]Marche Polytechnic University, Ancona, Italy, [4]LIMSI, CNRS,Orsay, France
[5]Idiap Research Institute, Martigny, Switzerland, [6]Fondazione Bruno Kessler, Trento, Italy

***Abstract*—This paper describes the speaker diarization systems developed for the Second DIHARD Speech Diarization Challenge (DIHARD II) by the *Speed* team. Besides describing the system, which considerably outperformed the challenge baselines, we also focus on the lessons learned from numerous approaches that we tried for single and multi-channel systems. We present several components of our diarization system, including categorization of domains, speech enhancement, speech activity detection, speaker embeddings, clustering methods, resegmentation, and system fusion. We analyze and discuss the effect of each such component on the overall diarization performance within the realistic settings of the challenge.**

***Index Terms*—Diarization, DIHARD challenge, evaluation, single-channel and multi-channel speech.**

I. Introduction

The DIHARD II diarization challenge [1] focused on “hard” diarization, by providing datasets that are challenging to the current state-of-the-art *speaker diarization* (SD) systems. The intentions of DIHARD II is to both (i) support SD research through the creation and distribution of novel data sets, and (ii) to measure and calibrate the performance of systems on these data sets. The challenge consists of four tracks, with two tracks for single channel data and two tracks for multi-channel data. The data for development and evaluation sets were taken from different databases, including audiobooks, meeting speech, child language acquisition recordings, dinner parties, and samples of web video. The organizers allowed using large set of existing speech corpora for training.

In this paper, we describe the efforts of the multi-national team *Speed* in the DIHARD II challenge. We focus on the approaches tried and lessons learned and present the system with considerably improved performance compared to the baseline provided by the challenge organizers. The main contributions of the *Speed* team can be summarized as follows:

- Automatic grouping of pseudo-domains for class-dependent SD was investigated.
- Different speech activity detection (SAD) methods were assessed.

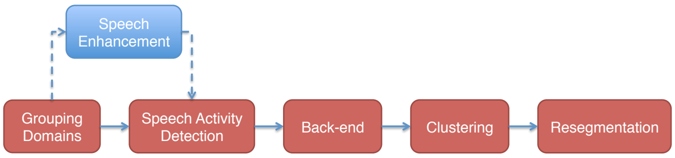

Fig. 1. Modules of *Speed* speaker diarization system.

- An in-house SD system was developed, outperforming the baseline provided by the organizers.
- Resegmentation methods are considered and approaches to their combination approaches are proposed.
- For multichannel data, the suitability of different front-end processing and clustering methods is considered in an attempt to improve the SD performance.

II. DIHARD II challenge

DIHARD II speaker diarization challenge [1] evaluates the task of determining “who spoke when” in a multi-speaker environment based only on audio recordings. As with DIHARD I [2], development and evaluation sets were provided but participants were free to train the systems on any proprietary or public data. DIHARD II extends the inaugural DIHARD I challenge by adding tracks on multi-channel recordings (from CHiME-5 [3]), by using more refined annotations for evaluation set, which made it more challenging for systems that over-fit on the development data, and by providing baseline system (the best performing system from DIHARD I [2]) to the participants. The DIHARD II has four different tracks. Two of them (Track 1 and 2) are dedicated for the single channel speech and other two (Track 3 and Track 4) are for multichannel. The difference between the two tracks in each channel is in the use of *speech activity detection* (SAD). Track 1 and Track 3 use ground-truth SAD labels provided by the organizers whereas the SAD labels need to be generated by the participants for the other two tracks.

DIHARD II uses two evaluation metrics. In addition to the previously used *diarization error rate* (DER) metric, a new metric called *Jaccard error rate* (JER) is introduced. The details about the DIHARD datasets and evaluation methodology are available in [1], [4].

⋆ Equal contributions.

This work was partly supported through funding from both the Agence Nationale de la Recherche (French research funding agency) and the Swiss National Science Foundation within the ODESSA (ANR-15-CE39-0010) and PLUMCOT (ANR-16-CE92-0025) projects.

TABLE I
SUMMARY OF THE DOMAIN GROUPING SHOWING THE NUMBER OF AUDIO FILES PREDICTED FOR EACH GROUP IN EVALUATION SET.

Group	Domains	# dev	# eval
1	audiobooks, broadcast interview, court room, & maptask	59	54
2	child language	23	38
3	clinical, sociolinguistic (field), & sociolinguistic (lab)	52	61
4	meeting, restaurant, & web video	58	41
Total	–	192	194

III. DIARIZATION SYSTEM

The speaker diarization system consists of several key modules as shown in Figure 1. In this challenge, we focus on enhancing several of them to improve the SD performance.

A. Grouping domains

The speech data for single channel tracks (Track 1 & 2) consists of speech files of 11 different domains such as audio books, web videos, broadcast interview, meetings, etc. These domains are different in terms of speech quality, number of speakers, recording environment, amount of overlapped speech, etc. Studies on speaker diarization with diverse domains indicate that domain-dependent processing helps in improving the overall performance [5]. Especially, this is important for selecting domain-dependent thresholds and for adaption of the *automatic speaker verification* (ASV) back-end. In order to exploit the advantages of domain-dependent processing, first we made an attempt to categorize the audio files according to the provided domain labels. We have developed an i-vector-based domain classification method but it shows poor classification accuracy even on development set. We have obtained 86.98% accuracy when tested with leave-one-out cross validation on the development set. The accuracy is poor not only because this is a challenging task but also due to the fact that some of the domains are similar to each other in terms of speech quality. Considering the fact that miss-classification of domains on unseen evaluation data can significantly decrease the diarization performance, we grouped the domains into a reduced number of classes. We use the confusion matrix of primary domain classification results, SD performance on individual domains in development set, as well as the metadata of the domains for this grouping task. Groups are summarized in Table I. This domain grouping also helps to increase the amount of audio-data for training class-dependent speech enhancement and SAD methods, which are described in the following two subsections.

B. Speech enhancement

On the front-end side, both for single-channel and multi-channel, we employed a *deep neural network* (DNN) based speech enhancement algorithm based on the deep feature loss paradigm [6]. The speech enhancement network is fully convolutional and takes the noisy raw waveform as input yielding the enhanced speech as output. We used a ResNet-inspired architecture [7] with squeeze-and-excitation blocks [8]. In each residual block, the dilated convolutional layer is followed by leaky *rectified linear unit* ReLU activations. In order to allow the network learning good inter-channel dependencies, a squeeze-and-excitation self-attention block with leaky ReLU activation and a dense layer with 16 neurons are used next. Skip connections have been implemented in our architecture, differently from [6]. Finally, we employed the same VGG-19 inspired network as in [6] to compute the loss. The Adam optimizer [9] and weight normalization [10] have been used for training.

Since no clean speech references are available in the DIHARD II Development dataset, we used synthetic datasets to train our network. We built four different synthetic datasets, one for each of the group of domains introduced in Table III-A. Each synthetic dataset was built to be as similar as possible to the corresponding group. We used clean speech utterances from Librispeech [11], ParlamentParla [12], ST Chinese Mandarin Corpus [13], and Freesound [14]. Other sources, like cry sounds, have been taken from Freesound and Youtube [15]. Noises have been extracted from the DIHARD Development set by using the reference SAD and used in conjunction with other noise sources like the MUSAN dataset [16] and Freesound. We generated the synthetic acoustic scene using Pyroomacoustics [17].

The speech enhancement algorithm has been validated by using the DIHARD II speech enhancement baseline as term of comparison and the synthetic datasets above as evaluation data. For each domain, a $70-20-10$ split for training, validation, and testing respectively has been used. *signal-to-noise ratio* (SNR) and the *perceptual evaluation of speech quality* (PESQ) were used as evaluation indexes. The results reported in Table II confirm the effectiveness of the proposed approach. We use this enhanced speech for training one of our SAD system.

TABLE II
PERFORMANCE COMPARISON (IN TERMS OF PESQ/OUTPUT SNR [DB]) OF PROPOSED AND BASELINE SPEECH ENHANCEMENT ALGORITHMS ON THE SINGLE CHANNEL SYNTHETIC DATASETS FOR THE FOUR GROUPS.

System	**Group 1**	**Group 2**	**Group 3**	**Group 4**
DIHARD baseline	2.92/8.44	2.70/5.49	2.69/5.14	2.73/4.97
Proposed algorithm	**3.12/9.02**	**2.80/5.94**	**2.80/5.95**	**2.79/5.84**

C. Speech activity detection

Speech activity detection (SAD) is modelled as a supervised sequence labeling problem. Let $\mathbf{x} \in \mathcal{X}$ be a sequence of feature vectors extracted from an audio recording (e.g., *mel-frequency cepstral coefficients* or MFCCs): $\mathbf{x} = (x_1, \ldots, x_T)$ where T is the length of the sequence. Let $\mathbf{y} \in \mathcal{Y}$ be the corresponding sequence of labels: $\mathbf{y} = (y_1, \ldots, y_T)$ and $y_i \in \{0, \ldots, K-1\}$ where K is the number of classes. In case of speech activity detection, $K = 2$ classes: $y_i = 1$ for speech, $y_i = 0$ for non-speech.

The objective is to find a function $g : \mathcal{X} \rightarrow \mathcal{Y}$ that matches a feature sequence $\mathbf{x}$ to the corresponding label sequence $\mathbf{y}$. We propose to model this function g using a stacked *long short-term memory* LSTM neural architecture trained with crossentropy loss. Short fixed-length sub-sequences (a few seconds) of otherwise longer and of variable length audio files are fed into the model. This allows to increase the number of training samples and augment their variability.

At test time, audio files are processed using overlapping sliding windows of the same length as used in training. For each time step, this results in several overlapping sequences of K-dimensional (softmax-ed) scores, which are averaged to obtain the final score of each class. The sequence of speech scores is then post-processed using two (θ_{onset} and θ_{offset}) thresholds for the detection of the beginning and end of speech regions [18].

In practice, half of DIHARD II development set was used for training, while hyper-parameters were tuned on the other half. We use an open-source implementation of this SAD approach as provided with the `pyannote-audio` toolkit [19].

D. Back-end

The DIHARD organizers provided a Kaldi-based x-vector back-end for speaker similarity measure. We developed a separate back-end that uses different acoustic features and parameters for neural network training.

1) Acoustic features: We use mel-frequency cepstral coefficients (MFCCs) as our primary acoustic feature. We extract 24-dimensional MFCCs using 24 filters. Unlike the implementation used in the baseline where window-based *cepstral mean normalization* (CMN) is performed, we apply utterance-dependent CMN where the global mean is computed from the speech regions. We have also investigated *inverted mel-frequency cepstral coefficients* (IMFCCs) which capture complementary information to MFCCs [20]. The IMFCCs are extracted in the same manner as MFCCs, except the warping scale is flipped to give more emphasize to the high frequency regions.

2) Classifier: We rely on an x-vector system that uses neural network for discriminative training. The neural network consists of the *time-delay neural network* (TDNN) architecture which captures information from large temporal context from the frame-level speech feature sequences [21]. In addition to the TDNN layers, x-vector system uses statistics pooling and fully connected layers to design a speaker classification network at segment level. In our x-vector implementation, we use five TDNN layers and three fully connected layers as used in [22]. The details of the neural network configuration is shown in Table III. The x-vectors are used with *probabilistic linear discriminant analysis* (PLDA) back-end for segment-level speaker similarity measure.

We have implemented the x-vector system with Keras Python library [23] using TensorFlow back-end [24]. We use (ReLU) [25] and *batch normalization* [26] for all the five TDNNs and two fully connected layers. We apply dropout with probability 0.05 only on the two fully connected layers. Parameters of the neural work are initialized with Xavier normal method [27]. The neural network is trained using the Adam optimizer [28] with learning rate 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and without decay. We train the neural network using speech segments of 1 s. The x-vector systems are trained with batch size of 100 and 10 epochs where each epoch consists of 6732100 mini-batches. We consider development sets of VoxCeleb1 and VoxCeleb2 consisting of 7205 speakers, with no data augmentation. We extract 512-dimensional speaker embedding from the output of FC1 layer (before applying ReLU and batch normalization).

TABLE III
DESCRIPTION OF THE LAYERS IN X-VECTOR ARCHITECTURE. #F STANDS FOR NUMBER OF FILTERS, KS FOR KERNEL SIZE, AND DR FOR DILATION RATE.

Layer	Details
TDNN-{1,...,4}	Conv1D (#F=1024, KS={5,3,3,1}, DR={1,2,3,1})
TDNN-5	Conv1D (#F=4096, KS=1, DR=1)
Statistics pooling	Computation of mean and standard deviation
FC-{1,2}	Fully connected layers (#nodes=512)
Softmax	Softmax layer with 7205 outputs

E. Clustering

The baseline clustering provided by the organizers is based on a rather simple yet effective implementation of the *agglomerative hierarchical clustering* (AHC). The clusters are created from the pair-wise similarity matrix of segment-level x-vectors. The optimal threshold to stop clustering is determined by minimizing the DER on the entire development set.

Starting from the observation that almost half of the overall DER is due to the missed speakers, we investigated alternative clustering strategies that might reduce the missed speaker rate, under the assumption that these errors are related to the presence of overlaps between speakers. Therefore, one first attempt was to allow overlaps between clusters and trying to close gaps between two segments assigned to the same cluster and separated by another speaker for less than 1 s. Unfortunately, this approach gave a noticeable deterioration on both sets by introducing higher false alarms. Another attempt was to revise the clustering process in a more traditional fashion where a left-to-right (L2R) clustering is performed first, followed by an AHC. The x-vectors are averaged on the segments generated by the L2R initial step. However, this did not improve the DER either.

F. Multi-channel front-end

For the multi-channel front-end, we investigated multiple speech enhancement methods. We investigate BeamformIt [29] where an enhanced signal is computed using a simple filter and sum beamforming technique and a combination of BeamformIt followed by the baseline speech enhancement method provided by the organizers [30].

We also investigated a source localization-driven source separation method, similar to [31] and detailed in [32]. The source location was estimated using the classical *generalized cross correlation* (GCC-PHAT) technique [33]. On obtaining the source location, a delay-and-sum (DS) beamforming is performed and a time-frequency mask corresponding to the localized speaker is obtained using a 2-layered bi-LSTM neural network using features obtained from the delay summed signal. Speech separation is done with speech distortion weighted multi-channel Wiener filter (SDW-MWF) [34] using the speech and noise covariance matrices obtained using the mask. The neural network to estimate the mask was trained using simulated data based on the WSJ0-2mix dataset [35]. The dataset contains a mixture of two clean WSJ utterances which we further reverberate using RIRs generated with CHiME-5 like microphone array geometry. Noise from the CHiME-5 dataset is also included in the simulate data to make it realistic. Another attempt to exploit the availability of multiple channel was to average x-vectors across the channels of each device.

G. Re-segmentation

The final module (see Figure 1) of the diarization system, for which we tested different approaches, is re-segmentation. Given the output of the clustering step, re-segmentation aims at refining speech segments boundaries and labels. Two different resegmentation methods were tested both separately and jointly. One is based on Gaussian Mixture models (GMMs) and another on long short-term memory (LSTM) recurrent neural networks.

a) GMM-based: A GMM is used to model every cluster hypothesized at the clustering step. The log-likelihood is calculated at feature level for every such GMM model. To counterbalance the noisy behavior of log-likelihoods at frame level, an average smoothing within a sliding window is applied to the log-likelihood curves obtained with each GMM cluster. Then, each frame is assigned to the cluster which provides the highest smoothed log-likelihood. This technique, previously used in [36], is expected to provide a finer boundary correction.

b) LSTM-based: Assuming that the output of the clustering step predicts N different speakers, this re-segmentation method uses the same principle as in Section III-C, but with $K = k + 1$ classes so that the $y_i = 0$ label is assigned for non-speech and every other $y_i = k$ are used for speakers k $\in \{1, ..., N\}$. At test time, and using the (unsupervised) output of the clustering step as its unique training file, the neural network is trained for a number of epochs E (tunable) and applied on the very same test file it has been trained on. The resulting sequence of K-dimensional scores is post-processed by choosing the class with maximum score for each frame. To stabilize the choice of the hyper-parameter E and make the prediction scores smoother, scores from the $m = 3$ previous epochs are averaged when doing predictions at epoch E. While this re-segmentation step does improve the labeling of speech regions, it also has the side effect of increasing false alarms (i.e., non-speech regions classified as speech). Therefore, its output is further post-processed to revert speech/non-speech regions back to the original SAD output. The technique was previously proposed in [37].

IV. Experimental results

In this section, we present the evaluation results of different approaches for the modules of the diarization system.

A. Comparison with baseline SD methods for Track 1

First, we compare the SD performance with the challenge baseline and our implementation for Track 1 which uses ground-truth SAD labels. The main differences between the baseline and our implementation are in the feature configurations and classifier back-end. We have trained the back-end system with smaller chunks of 1 s whereas the baseline system is trained with chunks of more than 4s. The baseline system also uses data augmentation from MUSAN and RIR datasets, and it uses domain adaptation by learning centering and whitening parameters from in-domain DIHARD data. Our system is simpler, since we neither use data augmentation nor domain adaptation. For our SD method, we use the same AHC as used in the Kaldi baseline system. The comparative results for Track 1 are shown in Table IV. The results indicate that our SD system is consistently better than Kaldi baseline for both development and evaluation data. We observe more improvement in JER metrics. For example, we obtain a relative improvement of 11.46% and 13.33% for development and evaluation sets respectively. We have not observed any improvement with PLDA adaptation. This is most likely because our system is trained with smaller chunks of 1 s, which already fit the DIHARD speech segments. On the other hand, for Kaldi baseline system, the domain adaptation might have helped to compensate the duration mismatch.

TABLE IV
PERFORMANCE COMPARISON (IN TERMS OF DER/JER IN %) OF THE X-VECTOR BASELINE AND IN-HOUSE IMPLEMENTATIONS ON TRACK 1.

x-vector system	Dev	Eval
baseline	23.70/56.20	25.99/59.51
in-house	**22.87/49.76**	**25.33/51.58**

B. Comparison with baseline SD methods for Track 2

In Table V, we have compared the performance of SDs with different implementation of SADs. The results indicate that our LSTM-based SADs yield consistent improvement over the WebRTC-based SAD provided as baseline for the challenge. We have obtained the best performance in evaluation set in terms of DER by using Kaldi baseline as back-end and LSTM SAD. Our SD system also shows lowest JERs in both development and evaluation set. Different versions of our LSTM-based SAD system correspond to different training and tuning strategies, such as (v1) using MUSAN database of noises for noise augmentation, (v2) using silences of Dev set for noise augmentation and the same set for training and tuning hyper-parameters, which expectedly led to the best performance on Dev set, (v3) splitting Dev set into two subsets for training and tuning the system and silences in Dev set for noise augmentation, and (v4) adding the enhanced speech to the training set. From the results, it can be noted that the best performing system on Eval set (DER metric) simply learned the specific data and speech annotations provided by the challenge. Although such approach improved the DER results, it may not generalize well to the other types of data.

TABLE V
PERFORMANCE COMPARISON (IN TERMS OF DER/JER IN %) OF THE X-VECTOR BASELINE AND IN-HOUSE IMPLEMENTATIONS ON TRACK 2.

x-vector system	Dev	Eval
baseline + WebRTC SAD	38.26/62.59	40.86/66.60
baseline + LSTM SAD (v1)	25.66/56.88	35.81/63.03
baseline + LSTM SAD (v2)	**25.01**/55.75	43.08/65.77
baseline + LSTM SAD (v3)	28.77/58.32	**33.02**/61.51
baseline + LSTM SAD (v4)	27.93/57.46	35.44/63.19
in-house + LSTM SAD (v3)	28.97/**53.99**	34.71/**58.21**

C. Impact of domain grouping

Table VI shows the results for domain grouping for both Track 1 and Track 2. We have computed the SD performance for both Kaldi baseline and for our system. We observe that the domain grouping improves SD performance compared to the condition without grouping. For example, the DER on evaluation set has been reduced to 24.25% compared to 25.99% of Kaldi baseline. JER is consistently lower for our system. However, DERs of Kaldi baseline is lower compared to our system on Eval set. This is most likely due to the wrong estimation of the domains in the Eval set.

TABLE VI
PERFORMANCE COMPARISON (IN TERMS OF DER/JER IN %) OF THE SPEAKER DIARIZATION X-VECTORS SYSTEMS WITH DOMAIN-GROUPING FOR TRACK 1 AND TRACK 2.

Track	x-vector system	Dev	Eval
1	baseline	23.03/53.38	**24.25**/56.04
	in-house	**22.83/49.41**	25.34/**50.75**
2	baseline	**28.65**/56.04	**32.60**/59.16
	in-house	28.68/**53.00**	34.39/**57.30**

D. Comparison of acoustic front-ends

In Table VII, we have compared the performance of MFCC and IMFCC acoustic front-end and found that IMFCC gives poorer SD performance compared to the MFCC. This is expected, since high-frequency regions of DIHARD data are more corrupted by the noise. However, we have found that a score-level fusion (with the weight optimized on the development set) of two systems improves the overall performance in all cases. The fused system is our best performing system for Track 1 for both DER and JER.

TABLE VII
PERFORMANCE COMPARISON (IN TERMS OF DER/JER IN %) OF MFCC, IMFCC AND A FUSED SYSTEM IN TRACK 1 USING THE IN-HOUSE X-VECTOR IMPLEMENTATION.

System	Dev	Eval
MFCC	22.87/49.76	25.33/51.58
IMFCC	25.47/51.30	27.43/53.33
Fusion	**22.85/48.62**	**24.72/49.95**

E. Effect of resegmentation

Table VIII presents the results obtained after applying the resegmentation techniques described in Section III-G on Track 2 for both development and evaluation sets. These techniques were applied on top of the clustering solutions generated by two systems. The first is based on the provided x-vector baseline, domain grouping as described in Section III-A and an LSTM-based SAD (v3). Each GMM and LSTM resegmentations result in a small but consistent decrease in DER by about 0.4% for the development set. However, when applying them jointly (an LSTM-based is followed by a GMM-based resegmentation), DER further drops to 27.87%.

Similar trend can be seen for the evaluation set. Even more so when LSTM and GMM resegmentations are applied jointly, effectively lowering DER to 31.03%, resulting in our best overall performing system for Track 2. When instead of the baselines, we use our in-house x-vector implementation with the same LSTM-based SAD (v3), we can notice similar trends of lower DER after the resegmentation is perfomed. However, resegmentation had a negative effect on performance in Track 1. While in Track 2, the missed speech detection that artificially splits same-speaker speech content into multiple clusters can be corrected by the resegmentation, despite the negative effect of the false alarms, it would not have such positive effect on Track 1, since the "oracle" SAD annotation is already provided.

TABLE VIII
PERFORMANCE COMPARISON (IN TERMS OF DER/JER IN %) OF THE RESEGMENTATION METHODS ON TRACK 2.

SD system	Method	Dev	Eval
x-vectors (baseline) + grouping+LSTM SAD	none	28.65/56.04	32.60/59.16
	GMM	28.25/**55.67**	32.02/**58.88**
	LSTM	28.21/57.01	31.41/59.60
	LSTM+GMM	**27.87**/56.77	**31.03**/59.22
x-vectors (in-house) + score fusion	none	28.77/**51.37**	33.75/**55.54**
	GMM	28.79/54.72	33.55/58.01
	LSTM	28.16/54.31	32.77/58.87
	LSTM+GMM	**27.86**/54.99	**32.37**/58.67

F. Experiments on multi-channel SD

We have performed SD experiments on the multi-channel tracks using the Kaldi x-vector system as back-end. The results for Track 3 are shown in Table IX. We observe that a better tuning of the threshold on the training set led to a small improvement in the system performance. We have also found that optimizing the threshold for each recording session gives a marginal improvement over baseline. For example, DER is reduced to 58.28% from 60.10% when "oracle" threshold is chosen for each session. Table IX also shows the results for two different clustering methods. However, the performances are considerably deteriorated for evaluation set. Most of this deterioration performance is due to an increase in the false alarm in contrast with a minor reduction of missed speaker and speaker confusion. We observe a large performance gap between development and evaluation sets which indicates that threshold optimized for the development set fail to generalize to the evaluation set.

We have also evaluated different beamforming strategies and speech enhancement methods for multi-channel scenario. Table X reports the results on Track 3 in terms of DER. In most cases, performance deteriorates in both development and

TABLE IX
SD PERFORMANCE IN TERMS OF DER % ON DEVELOPMENT AND EVALUATION SETS FOR VARIOUS APPROACHES IN TRACK 3.

System	Dev	Eval
DIHARD baseline	60.10	50.85
Baseline threshold tuning	60.01	49.97
Session-based oracle threshold	58.28	-
Bridge gap + overlap	62.63	56.61
L2R + AHC	60.20	61.27

evaluation sets. The alternative BeamformIt method slightly improves the performance on the development set but performs considerably poorer on the evaluation set. The best system is the combination of BeamformIt with the baseline enhancement which is just slightly worse than the baseline. Going more into details, applying the enhancement signals increases both the missed speaker rate and the speaker confusion. Averaging x-vectors over the four channels of a device does not result in any noticeable difference in the SD performance, probably because channels are very close to each other. Note that some methods were not investigated on the evaluation set due to poor performance or the end of challenge evaluation.

TABLE X
SD PERFORMANCE IN TERMS OF DER (%) ON DEVELOPMENT AND EVALUATION SET IN TRACK 3 FOR DIFFERENT FRONT-END PROCESSING. SLOC SDW REFERS TO THE LOCALIZATION DRIVEN SOURCE SEPARATION

System	Dev	Eval
BeamformIT	59.96	53.00
BeamformIT+enhancement	60.01	50.71
SLOC SDW	64.09	–
xvec averaging	60.04	–

V. LESSONS LEARNED AND FUTURE DIRECTIONS

From all the investigations and the experiments by the *Speed* team, we have found that the SD performance can be systematically improved by improving each module, including backend, SAD, resegmentation, and the combinations and fusions of different methods. From our work on this challenging realistic dataset, we noted several important issues that may be helpful to the community and that may require further investigation.

Domain grouping: The way we combine different domains in this work helps to improve the SD performance marginally. However, we have observed large intra-domain variability in terms of SNR, DER, number of speakers, etc. The available domain labels are mostly associated with audio sources than the individual speech quality. Possibly for this reason, the optimized thresholds for each domain are not considerably different. We hypothesize that the speech files need to be clustered according to the speech quality before performing class-dependent SD. This clustering could be helpful for multi-channel tracks also as the speech files are collected from different room reverberation conditions.

Domain adaptation for backend: With our current system, we do not observe improvement with simple PLDA adaption by learning centering and whitening parameters from the in-domain data. This contradicts with the results by the Kaldi implementation. We have speculated that the newly trained system already compensates the domain mismatch due to the duration variability. We plan to explore more advanced domain adaptation such as supervised domain adaptation and inter-dataset variability compensation.

Speech enhancement: The employment of data-driven based speech enhancement algorithms required the adoption of synthetic suitably labelled datasets for DNNs supervised training. The synthetic data generation task proved to be very challenging, and it surely deserve more attention, especially in terms of reducing the mismatch between synthetic data and the datasets used in the challenge. In particular, a special care should be devoted to modeling of the diverse non-stationary noise sources. From a more general perspective, the success of speech enhancement algorithms in speaker diarization systems inevitably passes through the adoption of well-matched models at back-end level. For instance, processing the speech material used for training the diarization system with enhanced speech is expected to improve the overall performance. All these aspects have not been adequately investigated by the *Speed* team so far, and they should be addressed in the future.

Threshold computation: The state-of-the-art SD system as used in DIHARD baseline optimizes the global threshold for speaker clustering on the development set and applies this threshold when computing the diarization labels for the evaluation set. This approach may not lead to the optimum performance due to the possible mismatch in both sets. Clustering audio recordings and applying cluster-wise threshold can be a tuning techniques that improves performance.

Robust feature extraction: We have observed that using different features conveying complementary information can improve the SD performance. However, the feature extraction process used in our system lacked additional processing that can improve the robustness. We plan to explore further the robust audio features speaker diarization.

VI. CONCLUSION

This paper summarizes the work done by *Speed* team for the second DIHARD challenge. We have discussed different methods explored for improving speaker diarization performance in realistic conditions. Amongst all the methods explored, we have found considerable improvement over baseline when using LSTM-based speaker activity detection methods. We have also discuss which approaches and enhancements that we tried did not work and speculate what could be the reasons for that. Diarization on different tracks of the DIHARD challenge turned out to be difficult not only due to the largely varying speech quality but also due to the wide mismatch between development and evaluation sets. We discuss some of the future directions which will be explored in a post-evaluation analysis.

References

[1] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du, S. Ganapathy, and M. Liberman, "The second dihard diarization challenge: Dataset, task, and baselines," in *Proc. Interspeech*, 2019.

[2] G. Sell, D. Snyder, A. McCree, D. Garcia-Romero, J. Villalba, M. Maciejewski, V. Manohar, N. Dehak, D. Povey, S. Watanabe, and S. Khudanpur, "Diarization is hard: Some experiences and lessons learned for the JHU team in the inaugural DIHARD challenge," in *Proc. Interspeech*, 2018.

[3] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. Interspeech*, 2018.

[4] N. Ryant, K. Church, C. Cieri, A. Cristia, J. Du, S. Ganapathy, and M. Liberman, "Second dihard challenge evaluation plan," *Linguistic Data Consortium, Tech. Rep*, 2019.

[5] M. Diez *et al.*, "BUT system for DIHARD speech diarization challenge 2018," in *Proc. Interspeech*, 2018, pp. 2798–2802.

[6] F. G. Germain, Q. Chen, and V. Koltun, "Speech denoising with deep feature losses," *arXiv preprint arXiv:1806.10522*, 2018.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE CVPR*, 2018, pp. 7132–7141.

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[10] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 901–909.

[11] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Proc. ICASSP*. IEEE, 2015, pp. 5206–5210.

[12] B. Külebi and A. Öktem, "Building an open source automatic speech recognition system for catalan," in *Proc. IberSPEECH 2018*, 2018, pp. 25–29. [Online]. Available: http://dx.doi.org/10.21437/IberSPEECH.2018-6

[13] Surfintech, "ST-CMDS-20170001 1 - Free ST Chinese Mandarin Corpus."

[14] Freesound. Freesound. [Online]. Available: https://freesound.org/

[15] YouTube. Youtube. [Online]. Available: https://www.youtube.com/

[16] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015, arXiv:1510.08484v1.

[17] R. Scheibler, E. Bezzam, and I. Dokmanić, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," in *Proc. ICASSP*. IEEE, 2018, pp. 351–355.

[18] G. Gelly and J.-L. Gauvain, "Minimum word error training of RNN-based voice activity detection," in *Proc. Interpseech*, 2015.

[19] pyannote contributors. pyannote-audio: neural building blocks for speaker diarization. [Online]. Available: https://github.com/pyannote/pyannote-audio

[20] S. Chakroborty, A. Roy, and G. Saha, "Improved closed set text-independent speaker identification by combining MFCC with evidence from flipped filter banks," *International Journal of Signal Processing*, vol. 4, no. 2, pp. 114–122, 2007.

[21] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328–339, March 1989.

[22] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: robust DNN embeddings for speaker recognition," in *Proc. ICASSP*, April 2018, pp. 5329–5333.

[23] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[24] M. Abadi *et al.*, "TensorFlow: large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[25] V. Nair and G. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010, pp. 807–814.

[26] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[27] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[28] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proc. ICLR*, 2015, pp. 1–15.

[29] X. Anguera, C. Wooters, and J. Hernando, "Acoustic beamforming for speaker diarization of meetings," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 7, pp. 2011–2021, September 2007.

[30] L. Sun, J. Du, C. Jiang, X. Zhang, S. He, B. Yin, and C.-H. Lee, "Speaker diarization with enhancing speech for the first dihard challenge," in *Proc. Interspeech*, 2018, pp. 2793–2797.

[31] Z. Chen, X. Xiao, T. Yoshioka, H. Erdogan, J. Li, and Y. Gong, "Multi-Channel Overlapped Speech Recognition with Location Guided Speech Extraction Network," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, Dec. 2018, pp. 558–565.

[32] Annonymous, "Analyzing the impact of speaker localization errors on speech separation for automatic speech recognition," in *IEEE Automatic Speech Recognition and Understanding Workshop (Submitted)*. Singapore: IEEE, Dec. 2019.

[33] C. Knapp and G. Carter, "The Generalized Correlation Method for Estimation of Time Delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, Aug. 1976.

[34] A. Spriet, M. Moonen, and J. Wouters, "Spatially pre-processed speech distortion weighted multi-channel Wiener filtering for noise reduction," *Signal Processing*, vol. 84, no. 12, pp. 2367–2387, Dec. 2004.

[35] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Proc. ICASSP*. IEEE, 2016, pp. 31–35.

[36] J. Patino, H. Delgado, and N. Evans, "The EURECOM submission to the first DIHARD Challenge," in *Proc. INTERSPEECH*, 2018.

[37] R. Yin, H. Bredin, and C. Barras, "Neural speech turn segmentation and affinity propagation for speaker diarization," in *Proc. Interspeech*, 2018.