# ID R&D System Description to VoxCeleb Speaker Recognition Challenge 2022

*Rostislav Makarov, Nikita Torgashov, Alexander Alenin,*
*Ivan Yakovlev, Anton Okhotnikov*

ID R&D Inc., New York, USA

{makarov,torgashov,alenin,yakovlev,ohotnikov}@idrnd.net

## Abstract

This technical report describes ID R&D team submissions for Track 1 (closed) and Track 2 (opened) for the VoxCeleb Speaker Recognition Challenge 2022 (VoxSRC-22). This year VoxSRC competition was focused on cross-age and same noise trials. In our solutions we used a fusion of deep ResNets and self-supervised learning (SSL) models trained on a mixture of private large dataset and publicly available VoxCeleb2 for Track 2, and a fusion of the same architectures trained on VoxCeleb2 only for Track 1. The final submissions achieved the first places on the VoxSRC-22 leaderboard for both Track 1 and Track 2 with a $minDCF_{0.05}$ of 0.088 and 0.062 respectively.

**Index Terms**: Speaker recognition, Speaker verification

## 1. System Setup

In this chapter, we will describe the training setup of neural networks that we used in the competition.

### 1.1. Architectures

As a main architecture we have chosen ResNet [1], that is widely used in speaker recognition [2], [3] and ECAPA-TDNN [4] trained on top of the features of self-supervised models, such as WavLM [5] and HuBERT [6].

#### *1.1.1. ResNet*

We used a ResNet-34 [7] architecture as a baseline and applied a couple of modifications to the original architecture that led to ResNets with 100 and 202 hidden layers. As inputs for ResNets we used Mel filter bank log-energies (MFB), with a 25 ms frame length, 10 ms step and the FFT size of 512 over 20-7600 Hz frequency limits. For all models in Track 1 we used 80 Mel filter banks, while for the Track 2 models we used 96 Mel filter banks for ResNet100 to increase the model's capacity, and 64 Mel filter banks for the ResNet202 due to the computational reasons. Finally, Frequency-wise Squeeze-Excitation (fwSE) [8] blocks with bottleneck size 128 were added to the end of each residual module. Details of the ResNet202 architecture are shown in the table 1.

#### *1.1.2. SSL + ECAPA-TDNN*

For SSL models, we followed same approach as presented in WavLM paper: stacked ECAPA-TDNN(C=1024) model on top of wav2vec-like architecture weighted features (outputs from all transformer layers and conv-extractor module). We used `facebook/hubert-large-ll60k` and `microsoft/wavlm-large` pretrained models from HugginFace transformers framework [9].

### 1.2. Loss function

We used AM-Softmax loss with the margin value set to 0.3 and a scale value set to 40 for all ResNet models in Track 2, while for SSL models training the AAM-Softmax loss was used with margin and scale parameters equal to 0.2 and 30 accordingly. For the Track 1 models training we utilized the AM-Softmax and reduced margin to 0.2 and scale to 35.

## 2. Self-VoxCeleb dataset

Inspired by the idea of the VoxCeleb2 dataset [10] collection, we adopted and modified the collection method to obtain a similar dataset of increased volume, to which we refer as a Self-VoxCeleb. The dataset size overcomes VoxCeleb2 dataset size by a multiple factor, and all the videos are licensed under the CC BY 4.0. We did not use any face recognition model and utilized a speech-based filtering only.

### 2.1. Collection scheme

In brief, the collection scheme was divided into 3 parts:

1. **Channel meta filtering.** Borrowing the main idea of searching speakers on YouTube from [10], we came up with a hypothesis, that there is a number of YouTube channels, that have predominantly one person speaking. For example, a person covering specific subjects like: DIY, unpacking, teaching, and so on. Mostly, such channels are easy to spot given only grid of video previews. We aggregated metadata for approximately 1 million channels, filtered it by minimal and maximal number of subscribers, channel topic (if presented) and some other attributes presented in meta.
2. **Channel videos selection.** We passed filtered channels to assessors, who's task was to pick up to 25 videos from each channel, that contain single speaking person based on the video preview.
3. **Audio segments filtering.** On the final stage, we applied a filtering of required speech segments. We extracted audio track from each video, and applied an embeddings extraction per 2 seconds segment without overlap using pre-trained SV model. We adopted an Hierarchical Agglomerative Clustering (HAC) over the extracted embeddings and saved the biggest cluster segments as a speaker representation. We also checked the similarities of clusters from different videos in one channel in order to maintain a good intra-speaker variability and to filter out clusters with noisy speaker. We also dropped some channels that had high cosine similarity score between the median channel embedding (duplicate speakers). We then sampled segments from the filtered clusters, and the speaker label was generated based on the channel id.

Table 1: *ResNet-202 architecture*

| **Layer name** | **Structure** | **Output** (C × F × T) |
|---|---|---|
| Conv2D | 3×3, 128, stride=1 | 128 × 64 × T |
| ResBlock-1 | $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \\ \text{fwSE}, [128, 64] \end{bmatrix} \times 6$ | 128 × 64 × T |
| ResBlock-2 | $\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \\ \text{fwSE}, [128, 32] \end{bmatrix} \times 16$ | 128 × 32 × T/2 |
| ResBlock-3 | $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \\ \text{fwSE}, [128, 16] \end{bmatrix} \times 75$ | 256 × 16 × T/4 |
| ResBlock-4 | $\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \\ \text{fwSE}, [128, 8] \end{bmatrix} \times 3$ | 256 × 8 × T/8 |
| Flatten (C, F) | — | 2560 × T/8 |
| StatsPooling | — | 5120 |
| Dense | — | 256 |
| AM-Softmax | — | Num. of speakers |

# 3. Experiments

### 3.1. Dataset

We used VoxCeleb2-dev dataset [10] for training the models for Track 1. For open condition (Track 2) we used both: Voxceleb2-dev (A) and Self-Voxceleb (B) datasets. For validation, VoxCeleb1-test [11] set and VoxSRC22 validation sets were used.

### 3.2. Data augmentation

For data augmentaion during the training we used MUSAN [12] and room impulse responses (RIR) [13] databases. For each training utterance, one of six various augmentation strategies was selected randomly:

- **Music**: A single music file is randomly selected from MUSAN and added to the original signal (5-15dB SNR). The duration of additive noise is matched to the duration of the original signal.
- **Noise**: Randomly selected noise from MUSAN added to the original recording (0-15dB SNR).
- **Speech**: Three to seven speakers are randomly picked, summed together, then added to the original signal (13-20dB SNR).
- **Reverb**: Artificially reverberate via convolution with real RIRs [14].
- **Speed**: We applied a speed augmentation that increased a number of speakers in training data by a factor of 3. [15].
- **SpecAugment**: We masked from 0 to 5 frames in the temporal axis and from 0 to 10 frames in the frequency axis using the SpecAug [16].

### 3.3. Initial training stage

All models were trained using TensorFlow 2 framework [17] on Google Cloud TPUs. For Track 1 we trained all models for 50 epoch, 5000 steps each. The batch size was set to 256, and 2-seconds segments were randomly cropped for each utterance in the batch. We have also scheduled values of learning rate and margin of AM-Softmax loss function. The learning rate scheduler has three phases: warmup, plateau and decay. The learning rate was increased linearly from 1e-5 to 0.1, while the margin was equal to zero, for the first 3 epochs in warmup phase. Then, the learning rate was fixed to 0.1 and the value of margin was linearly increased from 0 to 0.2 for the next 10 epochs in the plateau phase. After the margin achieved it's maximum value, the learning rate was decreased exponentially with a rate of 0.5 each 4 epochs in the decay phase. For the data augmentation we used strategies described in section 3.2.

For the open Track 2 we used similar training strategy with the following differences: the number of epochs was increased to 200, the length of random crop in the batch was increased to 4-second, the maximum value of margin was increased to 0.3. The scheduler of the learning rate was elongated: 8 epochs for the warmup phase, 32 epochs for the plateau phase and the period of decay phase was set to 15 epochs.

For the Track 1 we exploited only ResNet100 architecture, while for the Track 2 we used both ResNet100 and ResNet202 architectures. For SSL based models, we used similar training hyperparameters as for ResNets. We have trained multiple models with different hyper-parameters which are presented in the table 2.

### 3.4. Fine-tuning stage

At the fine-tuning stage we removed SpecAugment and Speed augmentations, and set L2-regularization to zero for all models. The value of margin was set to 0.3, and the value of scale was set to 30. The maximum value of learning rate was decreased to 1e-4 and the number of epochs was set to 20 and 50 for Tracks 1 and 2 accordingly. We also removed all remaining data augmentations for the Track 2 models while fine-tuning. For SSL based models we also unfreezed all the weights.

### 3.5. Pairwise scoring and AS-Norm

For inference, we sliced input samples (both enrollment and verification) into $10 \times 4$ seconds chunks resulting in 100 scores as shown in eqs. (1) to (3), the same way as it was done in [10] and [18].

$$N = 10 \tag{1}$$

$$N_{scores} = N \cdot N = 10 \cdot 10 = 100 \tag{2}$$

$$score = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} cosine(enroll_i, verify_j)}{N_{scores}} \tag{3}$$

Verification score was further normalized by utilization of an AS-Norm. The AS-Norm cohort included all VoxCeleb2-dev speakers (mean embeddings) with a *top N = 300* trials used to estimate mean and std of scores distribution for normalization.

### 3.6. Quality Measurement Functions

It is well known that Quality Measuring Functions (QMFs) give a huge performance boost, especially on VoxCeleb-based testing datasets [19][3]. We can describe such signal parameters as a quality of the signal, estimate the signal-to-noise ratio (SNR) or the strength of the reverberation using the RT60 metric. Recent speech quality researches [20][21] also attempted to predict quality scores such as Mean Opinion Score as well as noisiness,

Table 2: *Training hyper-parameters*

*RO - ResNet Opened track, RC - ResNet Closed track, SO - Self-supervised Opened track, FT - fine tuning stage.*

| Model index | Name | Details | Features | Pooling | Dataset | Loss margin and scale | Segment len (sec) | L2-reg | Augs | Batch size |
|---|---|---|---|---|---|---|---|---|---|---|
| RC1 | ResNet100 | [6,16,24,3] | MFB80 | Stats | A | AM, 0.2, 35 | 2 | $1e^{-5}$ | Y | 256 |
| RC1-FT | | | | | | AM, 0.3, 30 | 4 | N | Y | 160 |
| RC2 | ResNet100 | [6,16,24,3] | MFB80 | CAS | A | AM, 0.2, 35 | 2 | $1e^{-4}$ | Y | 256 |
| RC2-FT | | | | | | AM, 0.3, 30 | 4 | N | Y | 160 |
| RC3 | ResNet100 | [6,16,24,3] | MFB80 | Stats | A | AM, 0.2, 35 | 2 | $1e^{-4}$ | Y | 256 |
| RC3-FT | | | | | | AM, 0.3, 30 | 4 | N | Y | 160 |
| RO1* | ResNet100 | [6,16,24,3] | MFB96 | Stats | A,B | AM, 0.3, 40 | 4 | $1e^{-4}$ | Y | 256 |
| RO1-FT | | | | | | AM, 0.3, 30 | 6 | N | N | 160 |
| RO2-FT1** | ResNet100 | [6,16,24,3] | MFB96 | Stats | A,B | AM, 0.3, 30 | 4 | N | N | 256 |
| RO2-FT2** | | | | | | | 6 | N | N | 160 |
| RO2-FT3** | | | | | | | 8 | N | N | 128 |
| RO3-FT* | ResNet202 | [6,16,75,3] | MFB64 | Stats | A,B | AM, 0.3, 30 | 6 | N | N | 160 |
| RO4-FT | | | | | | | | | | |
| SO1-FT | WavLM ECAPA | ECAPA (C=1024) | WavLM-large | CAS | A,B | AAM, 0.2, 30 | 6 | N | N | 128 |
| SO2-FT | HuBERT ECAPA | ECAPA (C=1024) | HuBERT-large | CAS | A,B | AAM, 0.2, 30 | 6 | N | N | 128 |

* The Speed augmentation was not applied during initial training of this model.
** This model has higher proportion of Self-VoxCeleb in initial training stage compared to RO1 model.

coloration, discontinuity and loudness. Such models are trained on datasets labeled with human opinions scores, so these metrics assumed to be representative from a human's perception point of view.

Based on the available labels and development and train datasets we utilized the following QMF attributes for correction of the verification scores.

**Speech and total length** based QMF values were extracted with a help of a standard energy-based Voice Activity Detection (VAD) [22] module. After applying the VAD, we summed all the speech segments lengths into one value. List of generated and used in out submissions QMFs:

- $a$) speech length of the enrollment model file,
- $b$) speech length of the trial file,
- $c$) logarithm of sum of enrollment and trial files speech lengths,
- $d$) logarithm of sum of enrollment and trial files total lengths.

**Signal-to-Noise ratio** based QMF values were obtained using the same VAD module. After classifying the voiced and non-voiced segments of a signal, signal-to-noise ratio could be calculated using the following equation 4:

$$SNR_{dB} = 10 \cdot log_{10} \frac{P_{voice}}{P_{non-voice}} \quad (4)$$

where $P_{voice}$ and $P_{non-voice}$ are powers of voiced and non-voiced segments.

We used the following SNR values as QMF:

- $e$) SNR of enrollment model file,
- $f$) SNR of a trial file.

**NISQA** [21] Mean Opinion Score (MOS) was also used in Track 2 as a QMF term. It is an open-source model for non-intrusive speech quality estimation. NISQA predicts the human perception of a speech signal quality on a scale from 1 to 5. We utilized the NISQA output for the two following QMF values:

- $g$) NISQA MOS value of enrollment model file,
- $h$) NISQA MOS value of trial file.

We also tried to adopt the age detector based QMF scores, however we did not observe a performance improvement using that. Finally, all the QMF values were re-scaled to the range $[0, 1]$ using Min-Max normalization per attribute. For our final Track 1 submission we used QMF values $a) - f)$, and for the Track 2 $a) - d), g), h)$.

### 3.7. Evaluation protocol

System's performance evaluation was conducted using two following metrics:

- The minimum detection cost function used by the NIST SRE [23] with parameters $P_{Target} = 0.05$, $C_{Miss} = 1$ and $C_{FalseAlarm} = 1$.
- The Equal Error Rate (EER) which shows where False Acceptance (FA) and False Rejection (FR) error rates are equal.

Table 3: *Results on the VoxCeleb1-test and VoxSRC22 dev sets*

| Model | VoxCeleb1-O | | VoxCeleb1-E | | VoxCeleb1-H | | VoxSRC22 Dev | |
|---|---|---|---|---|---|---|---|---|
| | EER[%] | $DCF_{0.01}$ | EER[%] | $DCF_{0.01}$ | EER[%] | $DCF_{0.01}$ | EER[%] | $DCF_{0.05}$ |
| RC1 | 0.47 | 0.036 | 0.63 | 0.067 | 1.17 | 0.114 | 1.57 | 0.100 |
| RC2 | 0.45 | 0.039 | 0.65 | 0.069 | 1.19 | 0.116 | 1.62 | 0.099 |
| RC3 | 0.45 | 0.038 | 0.59 | 0.062 | 1.12 | 0.111 | 1.56 | 0.090 |
| RC1-FT | 0.44 | 0.030 | 0.56 | 0.063 | 1.07 | 0.105 | 1.45 | 0.089 |
| RC3-FT | 0.43 | 0.032 | 0.53 | 0.058 | 1.04 | 0.105 | 1.47 | 0.083 |
| RC2-FT | 0.36 | 0.037 | 0.55 | 0.060 | 1.05 | 0.104 | 1.42 | 0.088 |
| SO1-FT | 0.56 | 0.089 | 0.60 | 0.066 | 1.36 | 0.139 | 1.89 | 0.121 |
| SO2-FT | 0.49 | 0.071 | 0.59 | 0.071 | 1.30 | 0.135 | 1.68 | 0.108 |
| RO1 | 0.34 | 0.020 | 0.48 | 0.047 | 0.85 | 0.076 | 1.25 | 0.068 |
| RO2-FT2 | 0.20 | 0.012 | 0.42 | 0.041 | 0.80 | 0.076 | 1.16 | 0.065 |
| RO2-FT1 | 0.20 | 0.014 | 0.45 | 0.043 | 0.89 | 0.080 | 1.29 | 0.072 |
| RO2-FT3 | 0.20 | 0.017 | 0.42 | 0.040 | 0.80 | 0.076 | 1.15 | 0.066 |
| RO1-FT | 0.29 | 0.024 | 0.45 | 0.045 | 0.84 | 0.076 | 1.24 | 0.068 |
| RO3-FT | 0.14 | 0.019 | **0.33** | **0.035** | 0.68 | 0.063 | 0.96 | 0.059 |
| RO4-FT | **0.13** | **0.011** | 0.36 | **0.035** | 0.68 | 0.061 | 0.97 | 0.060 |
| **Fusion Close** | 0.35 | 0.036 | 0.53 | 0.056 | 1.02 | 0.100 | 1.33 | 0.083 |
| **Fusion Open** | 0.14 | 0.012 | 0.36 | **0.035** | **0.66** | **0.060** | **0.94** | **0.056** |

## 4. Fusion scheme and results analysis

The output of our system is an implementation of a linear fusion of cosine similarity scores for all the models and QMF values. To find the weights of each model in a score-level fusion we used the COBYLA optimizer on VoxSRC22-dev set. The trial score was obtained according to eq. (5):

$$S' = \begin{bmatrix} w_1 & w_2 & ... & w_n \end{bmatrix} \cdot \begin{bmatrix} S_1 \\ S_2 \\ ... \\ S_n \end{bmatrix} + \begin{bmatrix} v_1 & v_2 & ... & v_k \end{bmatrix} \cdot \begin{bmatrix} Q_1 \\ Q_2 \\ ... \\ Q_k \end{bmatrix} \quad (5)$$

where $w$ is a vector of models weights, $S$ is a vector of single models scores, $v$ is a vector of QMF weights and $Q$ is a vector of QMF values.

Our fusion and single models metrics for all the protocols of VoxCeleb1-test dataset and VoxSRC22 dev set are presented in the table 3. This table results already include pairwise scoring, AS-Norm and QMFs usage. From the results we can see, that the addition of Self-VoxCeleb dataset in training improves the metrics 20-50% relative, compared to the usage of VoxCeleb2-dev training dataset only (see $RC2 - FT$ and $RO2 - FT3$ submissions). Also, the optimal training hyperparameters for ResNets are not optimal for SSL based models training, and as a result we can not achieve the same quality. In our opinion, a grid-search of optimal hyperparameters would solve this problem. And lastly, from all of our used QMFs we have found that QMFs $c$) and $d$) had the highest weights in our fusion. We also noticed that increase of QMFs weights produces better metrics on VoxSRC-22 eval set. As a result, for our submissions after the optimal weights estimation on the VoxSRC-22 dev set, we linearly scaled the QMFs weights (cumulative QMFs weight increased from 10% to 30%).

## 5. Conclusions

In this report we presented our solutions for the Tracks 1 and 2 of the VoxSRC-22 challenge. We have found out a significant importance of usage of QMF values in fusion. We also observed a positive trend in extending the amount of training speech data for open Track 2, as our ResNet202 trained on a mixture of VoxCeleb2-dev and Self-VoxCeleb achieves state-of-the-art performance on the VoxCeleb1-test protocols. As a future work we would like to reach the supervised models quality with our SSL based models. We would also like to pre-train SSL models using a mixture of VoxCeleb2-dev and Self-VoxCeleb datasets.

# 6. References


[1] 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.

[2] J. Thienpondt, B. Desplanques, and K. Demuynck, "Tackling the score shift in cross-lingual speaker verification by exploiting language information," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7187–7191.

[3] M. Zhao, Y. Ma, M. Liu, and M. Xu, "The speakin system for voxceleb speaker recognition challange 2021," *arXiv preprint arXiv:2109.01989*, 2021.

[4] B. Desplanques, J. Thienpondt, and K. Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," *arXiv preprint arXiv:2005.07143*, 2020.

[5] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, 2022.

[6] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[7] D. Garcia-Romero, G. Sell, and A. McCree, "Magneto: X-vector magnitude estimation network plus offset for improved speaker recognition," in *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, 2020, pp. 1–8.

[8] J. Thienpondt, B. Desplanques, and K. Demuynck, "Integrating frequency translational invariance in tdnns and frequency positional information in 2d resnets to enhance speaker verification," *arXiv preprint arXiv:2104.02370*, 2021.

[9] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://www.aclweb.org/anthology/2020.emnlp-demos.6

[10] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.

[11] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[12] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[13] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5220–5224.

[14] I. Szoke, M. Skacel, L. Mosner, J. Paliesek, and J. Cernocky, "Building and evaluation of a real room impulse response dataset," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 4, p. 863–876, Aug 2019. [Online]. Available: http://dx.doi.org/10.1109/JSTSP.2019.2917582

[15] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Sixteenth annual conference of the international speech communication association*, 2015.

[16] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[17] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.

[18] H. S. Heo, B.-J. Lee, J. Huh, and J. S. Chung, "Clova baseline system for the voxceleb speaker recognition challenge 2020," *arXiv preprint arXiv:2009.14153*, 2020.

[19] J. Thienpondt, B. Desplanques, and K. Demuynck, "The idlab voxsrc-20 submission: Large margin fine-tuning and quality-aware score calibration in dnn based speaker verification," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5814–5818.

[20] C. K. Reddy, V. Gopal, and R. Cutler, "Dnsmos: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors," in *ICASSP 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6493–6497.

[21] G. Mittag, B. Naderi, A. Chehadi, and S. Möller, "NISQA: A Deep CNN-Self-Attention Model for Multidimensional Speech Quality Prediction with Crowdsourced Datasets," in *Proc. Interspeech 2021*, 2021, pp. 2127–2131.

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[23] Nist 2018 speaker recognition evaluation plan. [Online]. Available: https://www.nist.gov/system/files/documents/2018/08/17/sre18_eval_plan_2018-05-31_v6.pdf